THESE DE DOCTORAT DE l'UNIVERSITE PARIS 6

Spécialité :

Informatique

présentée par

Virginie EUDE

Pour obtenir le grade de

DOCTEUR DE l'UNIVERSITE PARIS 6

Sujet de la thèse :

Modélisation spatio-temporelle floue pour la reconnaissance d'activités militaires

soutenue le 17 Décembre 1998

devant le jury composé de :

Mme Bernadette BOUCHON-MEUNIER	Directrice de thèse
M. Gérard LIGOZAT	Rapporteur
Mme Sylviane SCHWER	Rapporteur
M. Emmanuel COLLAIN	Examinateur
Mme Anne DOUCET	Examinateur
M. Dominique FONTAINE	Examinateur
M. Michel GRABISCH	Examinateur

Remerciements

Mes remerciements s'adressent en premier lieu à Bernadette Bouchon-Meunier, ma directrice de thèse, qui est à l'origine de cette recherche. C'est elle en effet qui, pendant son cours de DEA, m'a permis d'apprécier la logique floue à sa juste valeur. Elle a ensuite accepté de m'accorder sa confiance et de me prendre sous son aile pendant trois ans, prodiguant conseils techniques et pratiques sur la démarche du chercheur. Je la félicite pour sa patience et sa ténacité à rabacher à des générations de thésards les règles d'or de la recherche. Je crois que c'est utile, en tous cas, moi ça m'a marqué, alors continuez !

En second lieu, je remercie Sylviane Schwer et Gérard Ligozat pour leur disponibilité et leur gentillesse. Ils ont su très vite se plonger dans mon manuscrit et en dégager des remarques judicieuses. J'espère leur avoir fait apprécier la logique floue, même si elle ne relève pas directement de leur domaine de compétences.

Je voudrais accorder une place d'honneur dans ces remerciements à Emmanuel Collain, qui m'a suivie durant ces trois années passées à Thomson. Il a su m'encourager et me diriger sur les bonnes pistes. Sans doute est-ce son expérience d'ancien thésard qui lui a permis d'accomplir aussi bien sa tâche. Merci pour les longues heures de relecture, parfois même dans les transports en commun, le stylo rouge à la main !

Je remercie vivement Michel Grabisch, Dominique Fontaine et Anne Doucet pour leur participation à mon jury. Je compte beaucoup sur Michel Grabisch pour m'aider à peaufiner mon chapitre sur les opérateurs d'agrégation. Mon travail repose en grande partie sur la méthode de comparaison de graphes élaborée par Dominque Fontaine, et je suis sûre qu'il saura se montrer un juge averti. Enfin j'espère qu'Anne Doucet ne m'en voudra pas trop de l'avoir attrapée dans un couloir du LIP6 pour lui infliger ce travail d'examinateur.

J'ai cotoyé durant ces troix années de recherche deux équipes qui ont toutes deux joué un rôle très important pour moi : l'équipe LOFTI et le service SEA. Les traditionnelles réunions LOFTI du jeudi m'ont beaucoup apporté, tant sur le plan technique que moral : merci donc à tous leurs animateurs anciens et présents. Bravo à Christophe Marsala qui supporte depuis si longtemps tous les petits problèmes d'organisation de l'équipe.

Merci à toute l'équipe SEA pour sa coopération et sa bonne humeur (avec une mention "seulement certains jours" pour Philippe Mavier !). Je n'oublierai pas les leçons de gestion de mémoire de François et Ludovic, de formattage de disque et autres opérations délicates de Yannick. Bon courage pour la suite à Thierry et Hervé, mes collègues de convention CIFRE.

Enfin, un grand merci bien mérité à Sylvain qui a subit toutes les conséquences désagréables d'une thèse : stress, vacances retardées pour cause de rédaction, et surtout les séances de relecture dans le métro.

Abstract

This search concerns military intelligence. We work here with information perceived on the theatre of operations, either by intelligence officers or by sensors, and with knowledge of the typical activities of the enemy. Due to the nature of the application, information can be incomplete and uncertain. The enemy never acts twice exactly the same way, so the description of his activity is vague.

The enemy's activities are modelled with hierarchical templates, giving the decomposition of principle activities into sub-activities. Each decomposition can have a different meaning, going from pure conjunction to pure disjunction. These activities are linked with different kind of constraints, among which temporal and spatial constraints, the most important ones.

Due to their vagueness, we represent these constraints with fuzzy sets. We translate the template into different fuzzy constraints graphs. One graph is used to model the temporal constraints : its nodes correspond to dates. This way we can express numeric and symbolic constraints between points or intervals. For the spatial constraints we have three different graphs, one for each type of spatial constraint considered : contraints on the relative positions, on the distances and on the directions. The nodes correspond to the coordinates of points.

The reasoning is based on the propagation of constraints. The optimal propagation is obtained with the calculation of the minimal graph, where the constraints are the most severe. The spatial case is more complex than the temporal one, and in this case we just have an approximation of the minimal graphs. But the reasoning is similar. Each observation on the battlefield gives new constraints that we propagate. If the minimal graph (or its approximation) is consistent, then the occurrence of the template is possible and we can predict the temporal or spatial characteristics of the events not yet observed.

We introduced heuristics to optimise the calculation of the minimal graph, our algorithm being based on the PC1 algorithm extended to fuzzy constraints. We introduced a new algorithm to deal with disjunctive constraints, economical in the fuzzy as well as in the non-fuzzy case. Disjunctive constraints augment the temporal expressive power and sometimes allow to merge two close templates.

Fuzzy constraints allow the refinement of the evaluation of the template's assumption, because they may be partially satisfied. A template is generally partially recognised, some sub-activities have not been observed. The global evaluation of templates must take into account first the satisfaction of the constraints and second the activities recognised and their interpretation in the decomposition.

Abstract

We introduce a few satisfiability measures for the first part of evaluation, consisting in the comparison between observed and theoretical constraints. For the second part of evaluation, we have to combine the different satisfiability degrees. We propose different aggregation operators with their interpretation in the case of templates. We extend the symmetric summations to give them more flexibility. And then, to build a template, the expert can use a decision tree to choose each operator according to different criteria.

The originality of the search lies in the introduction of fuzzy constraints to model scenarios in a goal of recognition. First fuzzy constraints allow to be closer of the real description of the scenario made by a human expert, description which is vague and expressed in natural langage. Second with fuzzy constraints we refined the evaluation of the hypothesis of scenarios and we are closer to the human reasoning. Most of the time a human can hardly decide between total satisfaction or no satisfaction at all.

Résumé

Le domaine d’application de cette recherche est le renseignement militaire. Nous travaillons à partir des informations perçues sur le théâtre des opérations, que ce soit par des officiers de renseignement ou par des capteurs, et avec une connaissance des activités "typiques" de l'ennemi. Du fait de la nature de l'application, les informations disponibles sont incomplètes et incertaines. De plus, l'ennemi n'agit jamais deux fois de façon totalement identique, donc la description de ses activités est vague.

Les activités ennemies sont modélisées par des "gabarits", arbres de décomposition des activités principales en sous-activités. Chaque niveau de décomposition peut avoir une interprétation différente, allant d'une conjonction pure à une disjonction pure. Dans l'arbre, les activités sont aussi liées par différents types de contraintes, parmi lesquelles les contraintes temporelles et spatiales qui sont les plus importantes.

Ces contraintes étant imprécises nous les représentons par des ensembles flous. Chaque gabarit est traduit sous la forme de plusieurs graphes de contraintes floues. Un premier graphe modélise les contraintes temporelles : ses nœuds sont des événements datés. Pour les contraintes spatiales nous utilisons trois graphes, un pour chaque type de contrainte : contraintes de positions relatives, de distances et de directions. Dans ces cas les nœuds des graphes correspondent aux coordonnées cartésiennes des points considérés.

Le raisonnement s'appuie sur la propagation des contraintes. La propagation est optimale lorsque l'on obtient les graphes minimaux, où les contraintes sont les plus restrictives possibles. Dans le cas de contraintes spatiales nous n'avons qu'une approximation des graphes minimaux, mais le raisonnement suit le même principe que pour les contraintes temporelles. Chaque nouvelle observation sur le théâtre des opérations fournit de nouvelles contraintes qui sont propagées. Si le graphe minimal (ou son approximation) est consistent, alors l'occurrence de l'activité modélisée par le graphe est possible et on peut même prédire les caractéristiques temporelles et/ou spatiales des éléments non encore observés.

Nous avons introduit des heuristiques afin d'optimiser le calcul du graphe minimal, notre algorithme étant fondé sur une extension de PC1 à des contraintes floues. Nous avons aussi introduit un nouvel algorithme pour le traitement des contraintes disjonctives. Il se révèle économique aussi bien dans le cas flou et dans le cas précis. L'intérêt des contraintes disjonctives est d'augmenter le pouvoir d'expression temporel et permet parfois de fusionner deux gabarits proches en un seul.

Résumé

Les contraintes floues permettent de raffiner l'évaluation des différentes hypothèses de scenarii en introduisant la notion de satisfaction partielle. Une activité est généralement incomplètement reconnue, certaines sous-activités n'ayant pu être observées, pour une raison ou pour une autre. L'évaluation globale doit prendre en compte d'une part la satisfaction partielle des différentes contraintes et d'autre part les activités reconnues et leur interprétation dans la décomposition hiérarchique de l'arbre.

Nous introduisons quelques mesures de satisfiabilité pour la première partie de l'évaluation qui consiste en une comparaison des contraintes observées avec les contraintes théoriques calculées par le graphe minimal. Pour la seconde partie de l'évaluation, il faut combiner les degrés de satisfaction obtenus. Nous proposons différents opérateurs d'agrégation utilisables, avec leur interprétation dans le cadre de gabarits. Nous étendons les sommes symétriques pour les rendre plus souples d'utilisation. Enfin, pour construire un gabarit, l'expert peut consulter un arbre de décision qui l'aide à choisir un opérateur d'agrégation particulier pour une décomposition donnée en fonction de différents critères.

L'originalité de la recherche est l'introduction de contraintes floues pour modéliser des scenarii à reconnaître. Les contraintes floues permettent dans un premier temps de reproduire plus fidèlement la description que l'expert fait du scénario, description imprécise et exprimée en langage naturel. Dans un second temps, l'évaluation des hypothèses de scenarii est beaucoup plus fine et plus proche du raisonnement humain. La plupart du temps en effet il est difficile de décider entre satisfaction totale ou non satisfaction.

Table des matières

I Introduction

Le problème de la reconnaissance d'activités militaires comporte beaucoup de similitudes avec la reconnaissance de scenarii en général. La reconnaissance de scenarii consiste en une comparaison entre la réalité observée et des scenarii prévisibles que l'on cherche à reconnaître. En général, on dispose d'une bibliothèque de scenarii envisageables et on essaie de reconnaître le ou les scenarii qui se déroulent réellement. On procède par élimination en détectant des incompatibilités entre la réalité et certains scenarii. Parmi tous les scenarii qui n'ont pas été éliminés, une des tâches de la reconnaissance peut aussi être de classer les scenarii, du plus crédible au moins crédible.

On effectue en reconnaissance d'activités militaires les mêmes types de raisonnements qu'en diagnostic de panne ou en diagnostic médical : on compare une série d'observations (évolution des paramètres ou des symptômes) avec la description de l'élément à reconnaître (panne ou maladie). Les techniques employées ici sont donc aussi applicables dans ces domaines.

Mais la reconnaissance d'activités militaires comporte aussi des spécificités liées au domaine d'application. Celles-ci sont exposées par la suite. L'emploi des techniques classiques de reconnaissance de scenarii ne permet pas de tenir compte de ces particularités. Ces techniques ne prévoient pas le traitement d'informations incomplètes et imprécises comme on peut en rencontrer dans le renseignement militaire ou dans d'autres domaines.

Tout le raisonnement en reconnaissance de scenarii est en fait axé sur la vérification et la propagation de contraintes. Un scénario est en effet une liste de faits contraints. Il ne suffit pas de reconnaître les faits tels quels, il faut aussi que ces faits vérifient les contraintes imposées. Certaines contraintes ont des répercussions sur les autres faits qu'ils est intéressant de prévoir pour anticiper la reconnaissance ou la non reconnaissance. La propagation des contraintes est le mécanisme essentiel qui permet cette anticipation. Les contraintes concernées sont essentiellement des contraintes temporelles et spatiales dans notre cas de reconnaissance. Nous présentons plus précisément en quoi consiste la reconnaissance d'activités militaires.

I.1 Motivation

Notre étude se situe dans le cadre des Systèmes d'Information et de Commandement (SIC). L'activité des SIC en général se résume à l'acquisition, la diffusion, le traitement, la présentation d'informations afin de faciliter la prise de décisions et le contrôle de leur exécution, au profit des commandements ou organismes militaires.

On retrouve dans ce domaine les applications suivantes :

- planification, suivi et conduite des opérations de forces intégrées des 3 armées (Terre, Mer, Air)
- renseignement et guerre électronique
- mise en œuvre de systèmes d'armement ou d'équipements des forces terrestres
- surveillance des frontières (ou zones sensibles)
- gestion des situations de crise

La reconnaissance d'activités militaires est un des objectifs du renseignement et intervient essentiellement dans les situations de crise. Une situation de crise nécessite la prise de décisions rapide et pour cela le système doit fournir à l'officier de renseignements une vue la plus claire possible de la situation.

La reconnaissance d'activités militaires constitue un cas très particulier de reconnaissance de scenarii. On dispose d'un certain nombre de connaissances sur le comportement typique d'un ennemi et on cherche à connaître son activité globale en exploitant des informations recueillies sur le théâtre des opérations. A partir de faits élémentaires perçus sur le terrain on essaie de reconstituer la réalité tactique et stratégique. L'objectif est de déduire assez tôt les intentions de l'ennemi afin de décider d'une réaction adaptée.

Les informations perçues peuvent être de différentes natures : mesures de capteurs (météo, radars, Elint pour tout ce qui est électromagnétique, Comint pour tout ce qui concerne les communications, Imint pour ce qui est issu du traitement d'images, GPS) ou observations humaines (Humint). Toutes ces données subissent un premier traitement et arrivent au système de reconnaissance sous une forme symbolique. Les informations parviennent donc sous la forme de faits élémentaires datés. Les besoins sont d'agréger ces données hétérogènes en une information de plus haut niveau.

Les deux aspects prépondérants pour la description d'activités militaires sont le temps et l'espace. En effet, des faits élémentaires comme un décollage d'avion ou une activité reconnaissance météo peuvent intervenir dans différentes activités, mais c'est essentiellement l'organisation de ces faits dans le temps et dans l'espace qui permet de distinguer l'activité globale à laquelle ils participent.

La particularité de la reconnaissance d'activités militaires par rapport à la reconnaissance de scenarii en général provient de la nature même des connaissances et des informations disponibles. Avant de décrire la nature de ces informations, nous rappelons les définitions de certains termes parfois ambigus.

On parle de valeur ***imprécise*** ou ***vague*** dans le cas d'une valeur mal définie, on ne peut en donner qu'une approximation. Cette approximation peut être sous la forme d'un ensemble flou, on parle alors de valeur ***floue***. Ces notions sont souvent liées à celle de ***granularité***, qui correspond au niveau de précision auquel on se situe.

Le terme ***incertitude*** recouvre lui une toute autre notion, il qualifie la confiance que l'on a en un fait ou une observation. On peut parler d'***événement incertain***, dans ce cas c'est l'occurrence même de l'événement qui est mise en doute, ou d'***événement de date incertaine***, ce qui signifie que l'occurrence de l'événement n'est pas remise en cause mais la date associée peut être fausse.

Les connaissances sur la façon d'agir d'un ennemi sont vagues. Les tactiques sont variables d'une armée à l'autre et une même armée n'applique jamais deux fois de façon totalement identique une même tactique. On ne peut décrire qu'un canevas des activités correspondant à une tactique et la reconnaissance doit englober tous les déroulements possibles de cette tactique. La nature des activités est à peu près fixée, mais c'est surtout l'organisation de ces activités dans le temps et dans l'espace qui est variable.

D'autre part, les informations obtenues sur le terrain sont incomplètes et incertaines. On n'est jamais certain de posséder une connaissance parfaite de l'activité ennemie. On ne peut pas conclure qu'une activité n'a pas eu lieu parce qu'on ne l'a pas perçue. Et les informations que l'on reçoit ne sont généralement pas certaines : les capteurs peuvent être brouillés ou fonctionner en mode dégradé.

Il est aussi difficile de combiner toutes les informations du fait de leur granularité variable : on ne traite pas de la même façon une information concernant un char et une autre concernant un régiment. Enfin, l'ordre dans lequel on reçoit les informations ne correspond pas forcément à l'ordre d'occurrence des événements : la reconnaissance doit donc être indifférente à l'ordre d'intégration des observations.

I.2 Technique des Gabarits

La technique des gabarits est un outil de fusion de données utilisé dans le domaine du renseignement militaire [Collain 95]. Le gabarit est un élément de base dans des systèmes d'aide à la décision chargés d'élaborer et d'évaluer des situations. Un gabarit représente un modèle d'une activité globale que l'on souhaite reconnaître.

Un système de reconnaissance d'activités militaires dispose d'une bibliothèque de gabarits et doit désigner le ou les gabarits qui correspondent le mieux à une situation réelle observée sur un théâtre d'opérations. Le système doit permettre de réagir au mieux et au plus vite, c'est pourquoi l'aspect prédictif est primordial. Il gère et ordonne les multiples hypothèses qu'il établit afin de présenter à l'officier de renseignements une vue de la situation la plus concise et la plus juste possible.

Comme nous le verrons en II.3.3 p. 67, la technique des gabarits emprunte des éléments à plusieurs mécanismes de resonnaissance temporelle. Elle présente en effet à la fois une partie reconnaissance structurelle, et une partie reconnaissance temporelle tout aussi importante. Les deux aspects sont essentiels, ce qui rend cette technique tout à fait particulière.

Notre recherche a pour but d'améliorer certains aspects de la technique de gabarits. Nous présentons ici cette technique telle qu'elle fonctionnait avant l'intégration des résultats de cette recherche. Ceci afin de dégager les limitations et les besoins du système. Nous allons présenter le modèle de décomposition structurelle d'un gabarit, la modélisation du temps, de l'espace et des contraintes opérationnelles et enfin l'évaluation et la gestion des différentes hypothèses.

I.2.1 Structure des gabarits

Les activités de l'ennemi sont modélisées par des structures hiérarchisées, donnant la décomposition d'activités principales en sous-activités. Les décompositions successives lient des événements simples et observables comme le *décollage d'un avion* à des activités plus complexes et abstraites comme une *attaque aérienne*.

Les gabarits sont le reflet de la connaissance de l’expert sur l’occurrence "normale" des activités. Ils décrivent les différentes étapes de la réalisation d'une activité particulière. Ces étapes sont elles-mêmes décomposées en étapes de plus bas niveau et ainsi de suite jusqu'à ce que l'on atteigne des activités élémentaires identifiables à des observations.

Les gabarits ont donc des structures d'arbres où la racine correspond à l'activité globale à reconnaître, les feuilles à des faits élémentaires observables et les nœuds à des activités intermédiaires. Les branches lient une activité aux activités qui participent ensemble à la réalisation de celle-ci. On distingue les activités atomiques, briques élémentaires correspondant à des faits observables (feuilles de l'arbre) et les activités composites (nœuds intermédiaires ou racine de l'arbre) regroupant plusieurs activités atomiques ou composites. Ce sont en fait des gabarits plus ou moins élaborés.

Lorsque des observations correspondent à certaines activités élémentaires décrites dans un gabarit, on crée une hypothèse de ce gabarit en remplaçant ces activités par les observations. On parle alors d'une instance de gabarit. Un gabarit est un modèle qui peut s’instancier au cours de la reconnaissance pour produire de nouvelles hypothèses qui vivent en fonction de l’association ou non de nouvelles données avec les nœuds.

La reconnaissance de la structure d'un gabarit est évaluée par un degré de reconnaissance. Elle peut se décomposer en deux opérations élémentaires. Au niveau atomique, on parle de corrélation de données élémentaires lorsque l'on associe une ou plusieurs données terrain à un élément de gabarit. Au niveau supérieur, lorsque l'on combine plusieurs éléments de nature différente d'un gabarit pour en déduire un objet plus abstrait ou plus complexe, il s'agit d'agrégation.

Un gabarit composite est l'association de plusieurs gabarits au moyen de conjonctions, disjonctions, itérations ou autres formes d'agrégation. On verra les différentes interprétations que l'on peut donner à ces agrégations en IV.3 p.149.

Voici un exemple simple de gabarit présentant deux formes d'agrégation :

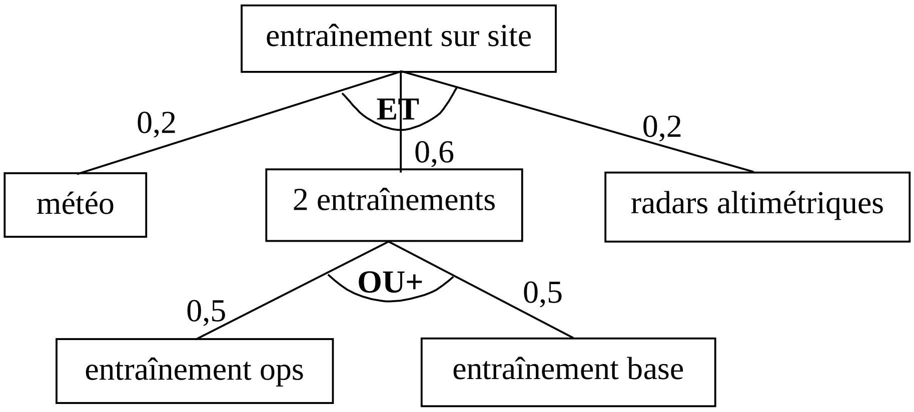


Fig. 1 : Gabarit entraînement sur site

Chaque sous-activité peut aussi être plus ou moins significative pour la reconnaissance de son activité de niveau supérieur, on attache donc des degrés d'importance aux activités. Une activité *reconnaissance météo* par exemple peut se retrouver dans de nombreux gabarits sans être caractéristique d'une activité particulière, elle sera donc associée la plupart du temps à un faible degré d'importance.

On reconnaît une activité *d'entraînement sur site* lorsqu'il y a la conjonction des activités *reconnaissance météo*, *deux entraînements* et *radars altimétriques*. L'activité *deux entraînements* est elle-même une agrégation d'un *entraînement ops* et d'un *entraînement de base* ; cette agrégation est en fait plus forte qu'une disjonction qui correspondrait à un "ou" logique. Une seule des deux activités suffit pour dire qu'il y a deux entraînements mais si on reconnaît les deux alors la certitude de l'activité deux entraînements est renforcée.

Un gabarit composant de plusieurs gabarits peut être exclusif. Les gabarits qui le contiennent sont alors exclusifs, c'est-à-dire que les hypothèses qu'ils modélisent ne pourront pas être valides simultanément.

Un gabarit ne doit pas rester une structure fixe s'il modélise des activités très proches mais distinctes. Ces variantes peuvent être exprimées par des éléments variables suivant le contexte rencontré. Un contexte est repéré par un élément de gabarit ou par une condition. Le choix d'un contexte spécifique fixe les contraintes et on se retrouve alors avec un gabarit classique.

Le gabarit *raid offensif* par exemple, suivant l'importance de l'opération, peut comporter ou non un *ravitaillement en vol*. Cette activité représente un contexte spécifique et on peut exprimer que l'activité de *navigation-tir* dure plus longtemps ([3h,4h]) s'il y a *ravitaillement* (cf. *Fig. 2*).

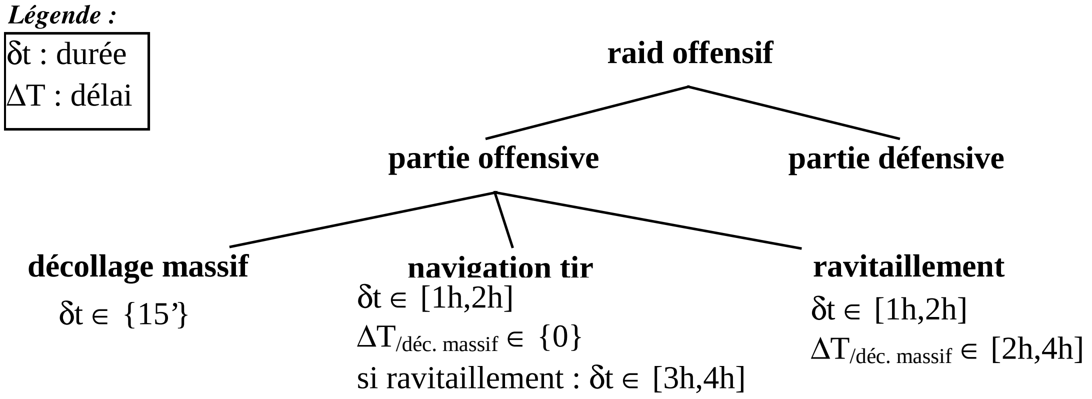


Fig. 2 : Gabarit raid offensif

I.2.2 Contraintes temporelles

On ajoute à cette décomposition structurelle des contraintes temporelles. Les contraintes temporelles peuvent être de deux natures. Un élément de gabarit peut posséder une contrainte quantitative sur sa durée. Cette contrainte est exprimée sous la forme d'un intervalle [durée minimale, durée maximale]. L'autre type de contrainte est un délai quantitatif relatif à un autre élément du gabarit. Ce délai peut être relatif au début du nœud père ou à la fin d'un des nœuds frères dans l'arbre du gabarit. Le délai est exprimé comme la durée par un intervalle [délai minimal, délai maximal]. Pour les phénomènes d'itération, il est possible de définir un delta temporel entre chaque itération.

Voici un exemple de gabarit avec ses contraintes temporelles :

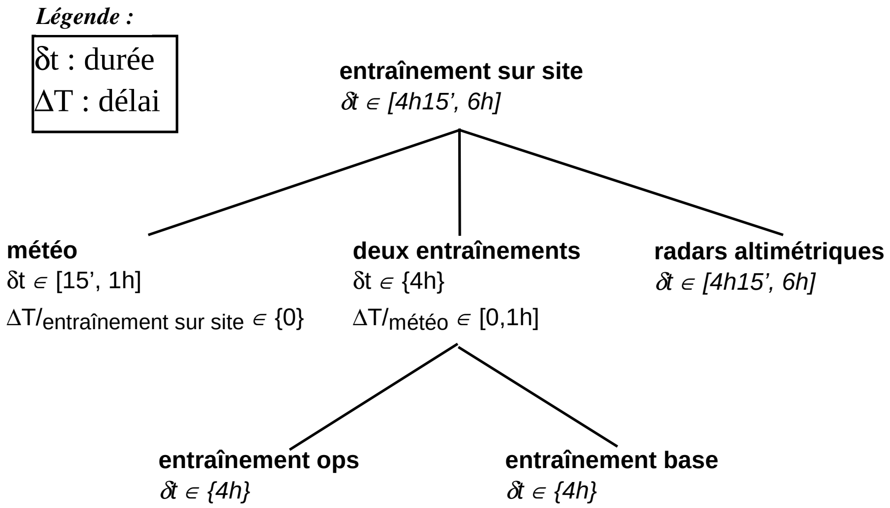


Fig. 3 : contraintes temporelles du gabarit entraînement sur site

Le raisonnement temporel consiste à associer à chaque gabarit atomique un intervalle daté correspondant à ses dates admissibles d'occurrence, appelé fenêtre temporelle. Par déduction à partir des contraintes temporelles contenues dans le gabarit et des dates réelles d'occurrence des éléments déjà reconnus, on met à jour les fenêtres temporelles de chaque élément du gabarit.

Soit e un élément de gabarit de fenêtre temporelle $[[d_1,d_2],[f_1,f_2]]$, ce qui signifie que e débute au plus tôt en d_1, au plus tard en d_2 et se termine au plus tôt en f_1 et au plus tard en f_2 et e' un élément de durée $[\delta_1,\delta_2]$ contraint par rapport à e. Deux cas peuvent se produire :

Cas 1 : e est un sur-élément de e' et le délai entre le début de e et le début de e' est $[\Delta_1,\Delta_2]$. Alors l'intervalle théorique de e' est $[[d_1+\Delta_1,d_2+\Delta_2],[d_1+\Delta_1+\delta_1,d_2+\Delta_2+\delta_2]]$. Il vaudrait mieux parler ici d'intervalle de recouvrement plutôt que d'intervalle théorique car certaines des valeurs de l'intervalle ne correspondent pas à des valeurs admissibles. On ne peut pas avoir une date de début valant $d_1+\Delta_1$ et une date de fin valant $d_2+\Delta_2+\delta_2$, sinon la durée n'est pas dans $[\delta_1,\delta_2]$. L'intervalle théorique correspond en fait à l'intervalle maximal des valeurs admissibles, mais tous les couples valeurs de début et de fin ne sont pas admissibles.

Cas 2 : e est un élément de même niveau que e' et le délai entre le début de e' et la fin de e est $[\Delta_1,\Delta_2]$. Alors l'intervalle théorique de e' est $[[f_1+\Delta_1,f_2+\Delta_2],[f_1+\Delta_1+\delta_1,f_2+\Delta_2+\delta_2]]$.

La mise à jour des fenêtres temporelles se fait en propageant ainsi les contraintes de proche en proche jusqu'à atteindre un état stable. Cette mise à jour intervient à chaque fois qu'une information sur la date ou la durée d'une activité arrive, c'est-à-dire lorsque l'on instancie un gabarit atomique par un nouvel événement, ou lorsqu'une précision supplémentaire est fournie sur un gabarit atomique déjà instancié.

Pour qu’une donnée soit associée à un gabarit atomique et que ce gabarit atomique soit reconnu, il faut non seulement qu’elle soit du type spécifié dans le gabarit mais ses dates d'occurrence doivent aussi être incluses dans la fenêtre temporelle du gabarit atomique. S'il y a instanciation, la fenêtre temporelle théorique est remplacée par la fenêtre temporelle réelle d'occurrence et il y a propagation.

I.2.3 Contraintes spatiales

On aimerait pouvoir raisonner de façon similaire dans le domaine spatial et dans le domaine temporel, en assimilant la géométrie ou le dimensionnement à une durée et la distance à un delta temporel. Mais les calculs dans un espace à deux, voire trois dimensions sont beaucoup plus complexes que pour le temps qui est linéaire. Tous les éléments ne peuvent donc pas être spatialement contraints par propagation.

Chaque donnée associée doit vérifier des contraintes spatiales locales au gabarit atomique auquel elle est attachée et des contraintes externes qui lient ce gabarit à d’autres. De même que l'on associait une fenêtre temporelle à chaque élément de gabarit, on lui associe un paramètre spatial qui est défini par sa géométrie (ponctuelle, linéique ...). Chaque contrainte doit être exprimée et vérifiée de façon explicite et il n'y a pas de mécanisme de propagation, contrairement au cas temporel.

On distingue le paramètre spatial théorique E^{Tg}, donné a priori sans tenir compte de contraintes externes, et le paramètre spatial réel E^{Td}, calculé après propagation des contraintes. La liste des dépendances $E^{T/}$ correspond à la liste des éléments qui ont des paramètres spatiaux en commun. Ici l'*entraînement sur site* a le même paramètre géométrique théorique E^{Tg} que les *entraînement ops* et *entraînement base*. L'accesseur spatial définit le type du paramètre spatial à considérer (position, zone…). On peut contraindre les paramètres spatiaux localement par une contrainte unaire E^{Cl} ou relativement à d'autres éléments par une contrainte binaire externe E^{Ce}.

Voici un exemple de gabarit avec ses contraintes spatiales :

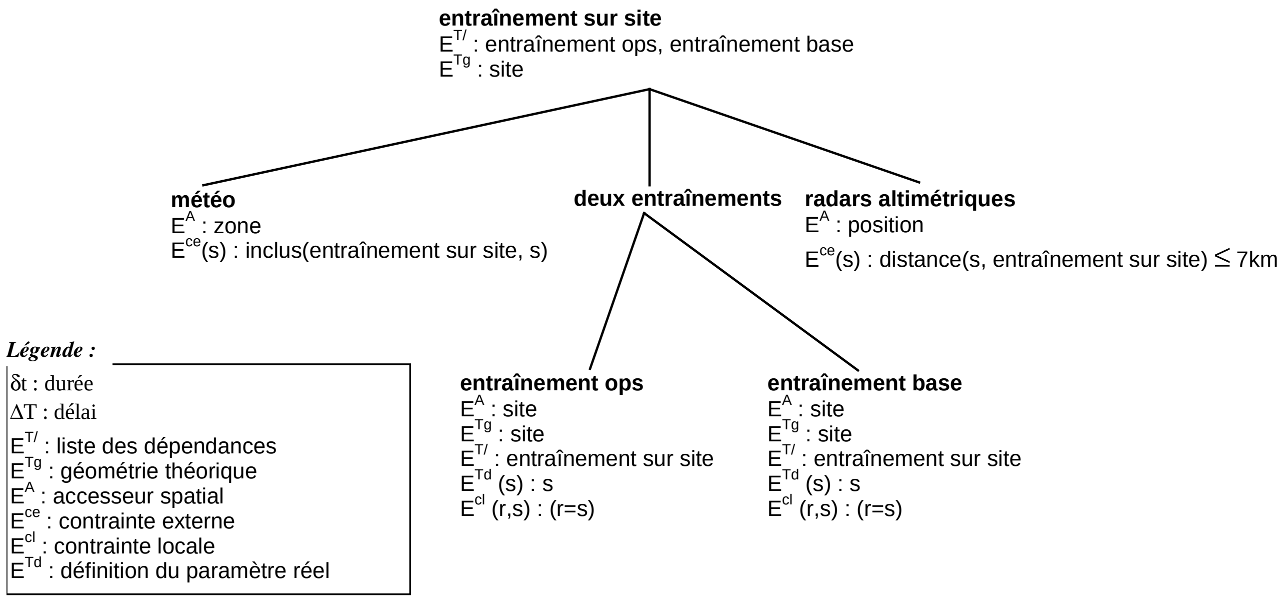


Fig. 4 : contraintes spatiales du gabarit entraînement sur site

On peut trouver différents types de contraintes spatiales. Les informations spatiales relatives aux observtions sont souvent des indications de positions données sous la forme de coordonnées géographiques. Comme on le voit sur le gabarit précédent, on peut aussi avoir des contraintes sur la distance entre deux points ou entre un point et une zone. On trouve aussi dans ce gabarit un exemple de contrainte topologique d'appartenance à une zone : le site de l'*entraînement* est inclus dans la zone de *reconnaissance météo*. Les opérationnels peuvent aussi introduire des directions relatives comme "au Nord" dans la description des gabarits ou dans les observations.

Pour représenter les aspects spatiaux d'un gabarit il faut donc utiliser des contraintes de différents formalismes mais ayant des interactions entre elles. Si deux points sont contraints par leurs positions la distance entre ces deux points est aussi contrainte. Ceci rend le raisonnement spatial beaucoup plus complexe que le temporel, et jusqu'ici on se contentait de vérifier les contraintes spatiales. L'introduction d'un mécanisme de propagation à ce niveau constitue un de nos objectifs.

I.2.4 Contraintes opérationnelles

D'autres paramètres peuvent être contraints, comme les variables *pilotes* qui peuvent avoir une contrainte sur l'appartenance à une certaine unité par exemple. La gestion de ces contraintes est similaire à celle des contraintes spatiales : on retrouve la définition du paramètre, des contraintes locales pour associer des données, des contraintes externes pour assurer la cohérence des différents éléments, d'un paramètre théorique et d'un paramètre réel.

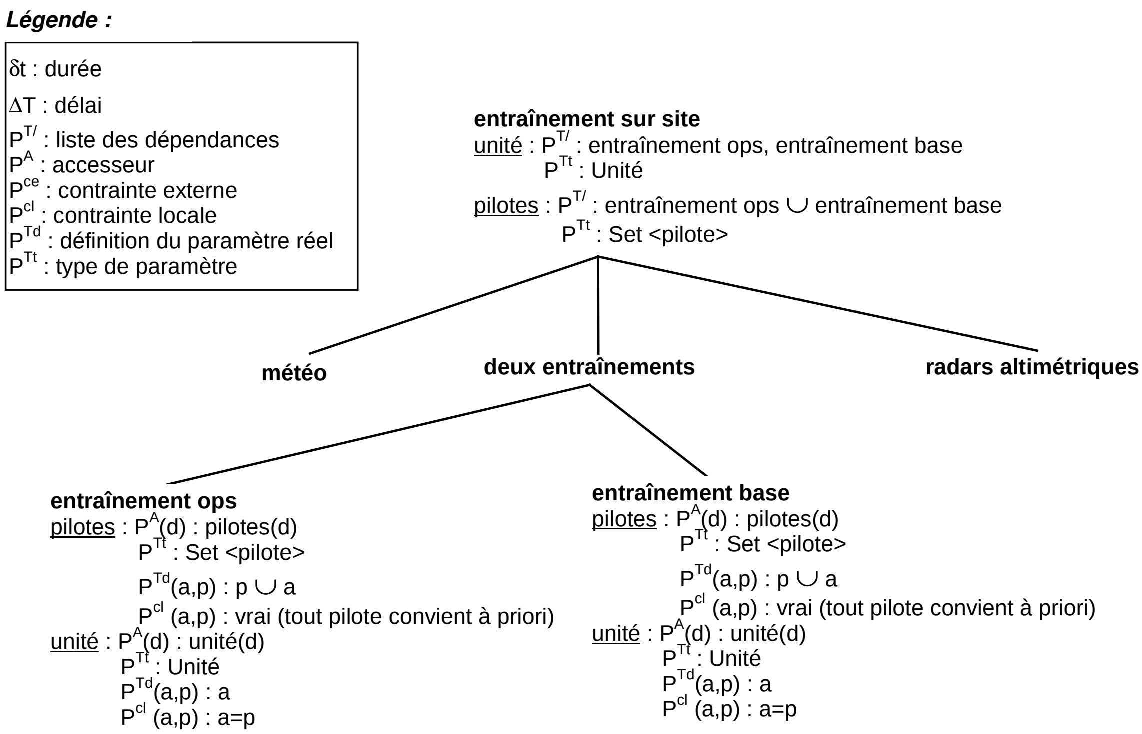


Fig. 5 : contraintes opérationnelles du gabarit entraînement sur site

I.2.5 Evaluation

Il s'agit d'évaluer les hypothèses de gabarits et de les ordonner en leur attribuant un degré de qualité. On attribue à chaque hypothèse de gabarit une mesure de certitude, sa nécessité. Celle-ci dépend du nombre et de la qualité des données associées à l'hypothèse en terme de respect des différentes contraintes. Pour un gabarit atomique, la certitude dépend de la corrélation des données reconnues ou parfois d'un paramètre lié aux données (pour un *décollage massif* par exemple, la certitude de cette activité augmente en fonction du nombre de décollages perçus).

De cette certitude on peut déduire une certitude sur le gabarit composite en fonction des types d'agrégation entre les éléments constituant le gabarit et de leur importance relative. Cette importance est représentée par une pondération de chaque élément éventuellement variable avec le contexte. Enfin, à chaque type d'agrégation est associé une formule traduisant sa nature (conjonction, disjonction, moyenne pondérée ...).

La certitude peut aussi augmenter avec le temps : elle augmente lorsqu'un fait prédit est reconnu ou diminue s'il ne l'est pas, et elle diminue aussi si aucune donnée associée à l'hypothèse n'a été reconnue depuis un certain temps. Plusieurs gabarits peuvent être dits "concurrents" : ils correspondent à des activités incompatibles dont le déroulement ne peut pas avoir lieu de façon simultanée. Pour les hypothèses concurrentes, si la certitude de l'une des hypothèses augmente alors celle de toutes les autres diminue et inversement. Cette opération est réalisée par une normalisation globale.

I.2.6 Cycle de vie

Une hypothèse peut naître, s’enrichir, s’appauvrir ou mourir au fil des événements. Pour chaque gabarit, certains éléments sont dits déclencheurs : ils sont caractéristiques du déroulement de l’hypothèse. Une hypothèse naît lorsqu'un ou plusieurs éléments déclencheurs sont reconnus et que le degré de certitude global du gabarit dépasse un certain seuil. Un élément extérieur au gabarit, qualifié de déclencheur exceptionnel, peut aussi avoir le même effet.

Dès qu’une hypothèse est déclenchée, toutes les contraintes temporelles se propagent dans l’arborescence du gabarit. Pour les hypothèses concurrentes, on maintient le plus longtemps possible toutes les hypothèses déclenchées en attendant un élément significatif pour choisir l’une ou l’autre.

A chaque nouvelle donnée associée, une hypothèse s’enrichit et ses contraintes sont mises à jour. Elle s’appauvrit si une donnée ne vérifie plus les contraintes du gabarit après une nouvelle mise à jour. A chaque mise à jour correspond aussi un nouveau calcul du degré de certitude.

Une hypothèse est abandonnée lorsqu’elle est invalide ou obsolète. Elle est invalide si elle atteint une certitude inférieure à un degré de validité depuis un temps supérieur au temps de validité. Le temps de validité correspond au délai, passé lequel il y a peu de chance que l'hypothèse s’enrichisse. Elle est obsolète lorsque l’on estime qu’elle n’est plus significative par rapport au contexte : elle ne peut plus évoluer. L’hypothèse devient aussi obsolète si sa certitude est supérieure au seuil de validité et qu’elle est active depuis un temps supérieur au temps d’obsolescence.

I.3 Objectifs

Le fonctionnement initial de la technique des gabarits tel qu'il vient d'être décrit fait ressortir un certain nombre d'inconvénients liés à la modélisation ainsi que des besoins pour aller plus loin dans le raisonnement. Un premier défaut de la méthode est une modélisation trop stricte des contraintes.

Les connaissances dans le domaine de la reconnaissance d’activités sont en effet très générales et approximatives. Les gabarits sont des schémas supposés classiques des activités que l’on souhaite reconnaître. Ils sont fournis par l’expertise et peuvent varier d’un expert à l’autre ou être fournis dans des termes vagues : "Normalement, lors d’un entraînement sur site, l’activité de *reconnaissance météorologique* doit durer entre quinze minutes et une heure."

Cette imprécision dans les termes employés est le reflet d'une réalité. Une activité ne se déroule jamais deux fois de façon totalement identique, et un gabarit doit être assez général pour reconnaître l'activité qu'il modélise quelles que soient les variantes de son déroulement. L'imprécision est présente essentiellement dans la description des contraintes, qu'elles soient temporelles ou spatiales.

Dans ces conditions, il est difficile de se limiter à une représentation des contraintes par intervalles numériques. Dans le fonctionnement actuel des gabarits, on constate en effet un manque de continuité et de finesse dans les résultats obtenus : une observation peut correspondre soit parfaitement soit pas du tout au modèle, mais il n'y a pas de position intermédiaire. Deux observations très peu différentes peuvent donc produire un résultat totalement opposé, ce qui, dans certains cas, a peu de signification étant donnée l'impossibilité de décrire précisément les modèles d'activité.

Tout ceci incite à modéliser les gabarits et à raisonner de façon plus souple. La nature des informations traitées correspond parfaitement à celles manipulées habituellement par la logique floue. Le choix de la théorie des ensembles flous semble donc parfaitement adapté pour la modélisation des contraintes. L'introduction de contraintes floues est un premier objectif de cette thèse.

L'introduction de ces contraintes floues oblige à modifier la procédure d'évaluation des gabarits. Au lieu d'avoir des contraintes totalement ou pas du tout satisfaites, on devra considérer des contraintes partiellement satisfaites. Ceci implique de revoir les deux types d'opérations qui constituent l'évaluation : la corrélation et l'agrégation. Un autre objectif de cette recherche est de traiter les contraintes spatiales au même titre que les contraintes temporelles. Cet objectif est lié à l'application.

Les objectifs de l'étude peuvent donc se résumer en trois points :

1. **une modélisation floue du temps et de l'espace**
2. **la réalisation d'un algorithme de propagation formel et général qui puisse s'adapter aussi bien à des contraintes temporelles que spatiales**
3. **une évaluation des hypothèses qui tienne compte de l'imprécision des descriptions**

Le point 2 met l’accent sur les contraintes temporelles et spatiales, nous allons en détailler les raisons. Pour l'instant, les gabarits ont principalement été utilisés pour de la reconnaissance d'activités, application où le temps était le critère prépondérant. Mais on envisage aussi de les utiliser pour de la reconnaissance plus orientée vers des critères spatiaux.

Dans le cadre actuel, les gabarits constituent des schémas classiques d’activités militaires. Ils se décomposent en sous-activités élémentaires observables. Pour reconnaître un certain scénario modélisé par un gabarit, il ne suffit pas de percevoir les activités élémentaires qui le composent, mais l’ordonnancement et les durées des différentes activités ainsi que les délais qui les lient doivent aussi être respectés.

Les gabarits doivent permettre de distinguer des scenarii comportant des événements assez semblables mais se déroulant différemment dans le temps, ce qui peut modifier totalement leur signification. Le temps joue donc un rôle discriminant primordial dans ce type de reconnaissance et il doit apparaître comme un paramètre essentiel.

On peut en fait imaginer un deuxième type de gabarit qui ferait intervenir l'espace comme paramètre le plus discriminant. On envisage en effet d'utiliser la technique des gabarits pour reconnaître les forces présentes sur le terrain : ceci consiste à distinguer les différentes unités et leur organisation. C'est dans ce cas le paramètre spatial qui est déterminant pour repérer suivant sa position si un élément appartient à telle ou telle unité. L'espace joue ici le même rôle que le temps dans le cas précédent de reconnaissance d'activités. Enfin, dans certains gabarits le temps et l'espace ont en parallèle un rôle discriminant.

Le temps et l'espace sont donc susceptibles de jouer des rôles similaires dans les gabarits. Ceci implique d'uniformiser le traitement des deux types de contraintes. Jusqu'à présent on vérifiait seulement que les contraintes spatiales étaient satisfaites, mais il apparaît indispensable d'effectuer de la propagation comme cela se fait pour les contraintes temporelles. Ceci justifie le deuxième objectif de cette thès,e l'introduction de raisonnements similaires pour les contraintes spatiales et temporelles.

La réalisation de cet objectif passe nécessairement par une formalisation du raisonnement. La propagation des contraintes temporelles est actuellement obtenue par un algorithme qui présente deux inconvénients majeurs : d'une part il ne comporte pas de justification théorique et on ne peut prouver ni qu'il se termine, ni qu'il propage complètement toute l'information, et d'autre part il est très lié à la structure actuelle des gabarits. Cet algorithme impose l'utilisation exclusive de contraintes de durées ou de délais telles qu'elles ont été définies en I.2.2 p.6.

I.4 Plan de l'étude

La thèse se décompose en quatre parties principales. La première partie est une synthèse bibliographique des différentes méthodes de reconnaissance de scenarii temporels, elle comporte à la fois un aspect modélisation temporelle et un aspect technique de reconnaissance. Nous avons vu en effet que la reconnaissance d'activités militaires pouvait être assimilée à de la reconnaissance de scenarii. Nous avons mis l'accent sur les scenarii temporels qui sont la seule forme de scenarii contraints que l'on trouve en abondance dans la littérature.

Dans une deuxième partie, on décrit la modélisation et le raisonnement retenus pour l'aspect temporel. On combine une modélisation par des FCSP (Fuzzy Constraints Satisfaction Problems) et une méthode de reconnaissance par comparaison de graphes. Le temps étant modélisé de façon linéaire, c'est l'aspect le plus simple à traiter et c'est pourquoi l'aspect temporel de la reconnaissance est présenté en premier lieu.

La partie suivante concerne l'évaluation des hypothèses. L'originalité essentielle pour cette évaluation est la prise en compte de satisfactions partielles de contraintes. On présente des mesures de corrélation et d'agrégation utilisables pour les gabarits. On a ainsi une vision globale du déroulement de la reconnaissance pour des scenarii temporels.

Enfin, dans la dernière partie, après une synthèse sur les modélisations spatiales, on montre comment on peut adapter le raisonnement de reconnaissance par comparaison de graphes au cas spatial. La synthèse sur les modélisations spatiales permet de cerner les possibilités de modélisation qui sont offertes dans ce domaine afin de remplir au mieux les besoins des gabarits. Avec les contraintes spatiales, l'algorithme est plus complexe mais le principe général reste le même.

II Reconnaissance de scenarii temporels

Dans ce chapitre nous présentons les techniques de reconnaissance de scenarii temporels les plus courantes, avec les avantages et les inconvénients de chacune et une comparaison avec la Technique des Gabarits. Ces techniques sont fondées sur différentes modélisations temporelles et utilisent des raisonnements temporels distincts. Pour bien comprendre ces techniques nous allons donc présenter d'abord les modélisations temporelles susceptibles d'être utilisées pour la reconnaissance, ainsi que les raisonnements qu'elles permettent.

Les modélisations sont classées suivant la nature des relations temporelles : celles-ci peuvent être symboliques, numériques, symboliques et numériques (mixtes). Les relations peuvent aussi être représentées par des ensembles flous. Pour une même modélisation, on peut trouver plusieurs catégories de raisonnements. On trouve des raisonnements qui sont plutôt des approches de type CSP (Constraints Satisfaction Problem), des approches géométriques ou autres comme nous allons le voir. Suivant l'approche que l'on choisit, le raisonnement est plus ou moins complexe et on peut répondre à certaines questions mais pas à d'autres. Ceci dépend de la complétude des approches.

II.1 Modélisations temporelles

Les notions temporelles prennent un caractère bien particulier dans de nombreux domaines de l'IA : planification, reconnaissance de situations, contrôle de processus, compréhension du langage naturel... Il s'avère nécessaire, lorsque l'on raisonne sur le temps, de le représenter explicitement et de le traiter de manière spécifique. C'est pourquoi la logique classique est insuffisante pour les raisonnements temporels : le temps y intervient comme simple argument d'un prédicat, et il est difficile et très coûteux d'exprimer ainsi toute sa spécificité.

On peut considérer deux niveaux de traitement du temps [Haton 91] : soit on s'intéresse à la gestion, à la mise à jour de données temporelles (gestionnaires de graphes temporels ou Time Map Manager, TMM), soit on cherche à raisonner sur les événements et les aspects de causalité (Temporal System Analyzers, TSA).

Ici notre intérêt se porte plutôt sur la première catégorie de problèmes puisque l'on raisonne sur les données temporelles indépendamment des événements qui leur sont attachés. On ne présentera donc ni les différentes logiques modales [Bestougeff 89], ni les logiques réifiées [Ghallab 92] ni les logiques temporelles possibilistes [Dubois 91], [Steunou 92].

Nous passons en revue les différentes modélisations temporelles intervenant dans des gestionnaires de graphes temporels. Ces modélisations se distinguent suivant qu'elles adoptent une approche symbolique, numérique, mixte ou floue et que les représentations sont ponctuelles ou par intervalles. Enfin nous aborderons les différentes techniques employées pour gérer ces données temporelles.

II.1.1 Représentations symboliques

La réification est un terme général pour signifier que des formules de la logique classique sont traitées comme les objets d'un langage plus général. Les logiques réifiées consistent à traiter des couples <relation logique ; notion temporelle> comme les nouveaux éléments d'une logique globale [Ghallab 92].

On distingue deux grandes théories temporelles symboliques dites logiques réifiées : la théorie d'Allen où l'élément de base est l'intervalle et celle de Mc Dermott fondée sur la notion d'instant. Ces deux théories donnent naissance à deux représentations distinctes des données temporelles. Ces représentations ont été abondamment utilisées indépendamment des logiques réifiées auxquelles elles sont attachées.

A) Théorie d'Allen

La logique d'Allen est une logique du premier ordre réifiée [Ghallab 92] où les primitives temporelles sont des intervalles et les relations sont symboliques [Haton 91]. Le temps est infini (pas de premier ni de dernier intervalle) et linéaire. Allen considère en effet que raisonner dans le futur équivaut à créer une instance d'hypothèse [Allen 84].

Allen définit 13 relations binaires entre intervalles qui forment la base de la représentation [Allen 81]. Ces relations sont complètes et mutuellement exclusives : à une situation donnée correspond une et une seule disjonction de relations primitives.

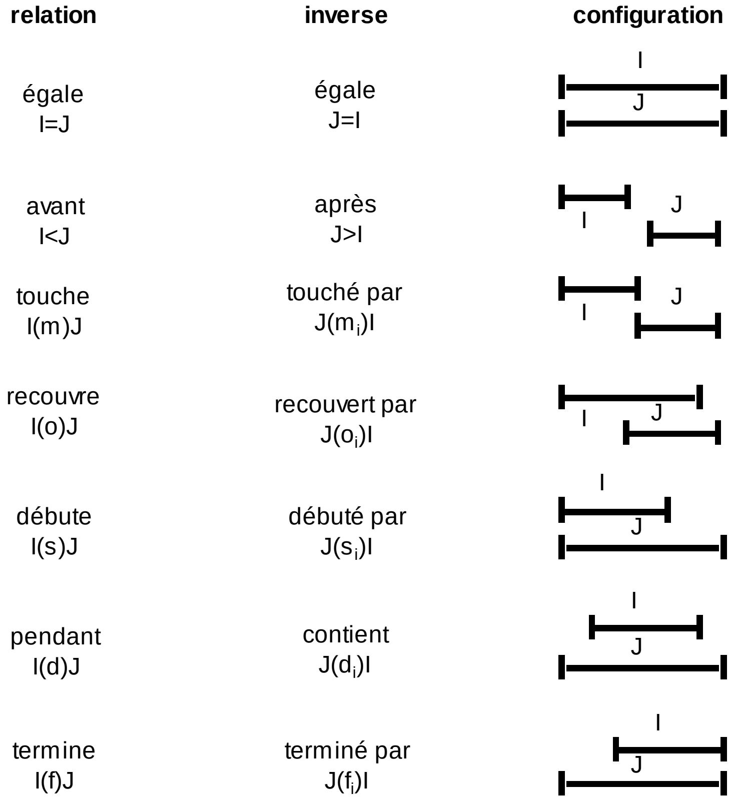


Fig. 6 : primitives d'Allen

Les nœuds des graphes temporels sont les intervalles. Entre deux intervalles, il existe toujours une disjonction de relations primitives qui est l'étiquette de l'arc orienté les reliant. Lorsque l'on a aucune information sur la relation entre deux intervalles, on les lie par la relation universelle qui est la disjonction des treize relations primitives.

L'intérêt d'un tel système est d'autoriser les imprécisions par des relations uniquement symboliques ainsi que les incertitudes grâce aux disjonctions. Pour modéliser la notion de présent, on crée un intervalle spécial "now"qui évolue dans le graphe sans que les autres relations ne soient modifiées. On peut exprimer la notion de persistance d'un fait en lui attachant un intervalle de persistance qui ne s'interrompt que lorsqu'il rencontre un intervalle où le fait s'infirme. Pour raisonner sur plusieurs hypothèses de relations entre intervalles comme "A avant B ou avant C", une solution est de lier le système à un ATMS (Assumption Truth Maintenance System) ou de traiter plusieurs problème distincts comportant une hypothèse différente sur les disjonctions. Un problème aura la contrainte "A avant B" et l'autre "A avant C".

On peut décrire la situation suivante par un graphe dans le formalisme d'Allen [Van Beek 92] : *"Fred lisait le journal pendant son petit déjeuner. Il posa le journal et finit son café. Après le petit déjeuner, il est parti se promener."* On peut trouver différentes versions de cet exemple suivant l'interprétation que l'on fait du terme "pendant" qui est ambigu. Ici on suppose que l'intersection entre l'action de *lire le journal* et celle de *prendre le petit déjeuner* est un intervalle non réduit à un point. Toutesl les contraintes qui vérifient cette condition sont acceptables.

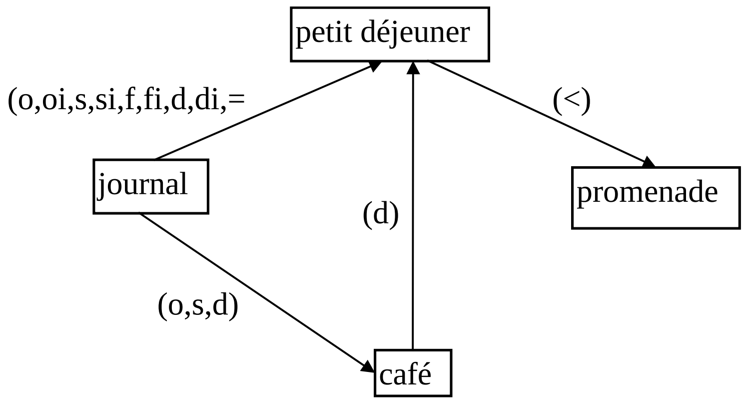


Fig. 7 : représentation dans le formalisme d'Allen

B) Théorie de Mc Dermott

Les notions introduites par Mc Dermott sont assez proches de celles d’Allen. La principale différence vient de l’utilisation de primitives temporelles ponctuelles. Mc Dermott raisonne lui sur des instants et non plus sur des intervalles. De plus, le temps est linéaire dans le passé et branché dans le futur. Il est aussi dense et infini. La logique de Mc Dermott est néanmoins, comme celle d’Allen, une logique temporelle réifiée du premier ordre [Haton 91] et [Mc Dermott 82]. Les instants possèdent une relation d’ordre partiel ≤.

Le principe de la représentation est le même que dans le cadre de la théorie d'Allen, en remplaçant tous les éléments de l’algèbre d’intervalles par les éléments correspondants de l’algèbre d’instants. Les nœuds du graphe sont des points, les primitives sont les relations <, >, et = et la relation universelle est (< ou > ou =).

De même que dans les graphes d’Allen, on ne peut pas envisager plusieurs hypothèses du type : I avant J ou I avant K.

La représentation a été simplifiée dans IxTeT (**I**nde**x**ed **T**im**e** **T**able) [Mounir-Alaoui 90], [Ghallab 89.1] afin d’augmenter l’efficacité des traitements (cf. II.2.3). Il n’y a plus que des arcs étiquetés par ≤ et par ≠. Toutes les relations de l’algèbre d’instants peuvent s’exprimer avec 0, 1 ou 2 arcs étiquetés par ces relations.

C) Comparaison des deux approches

En fait, la plupart des relations de l'algèbre d'intervalles peuvent s'exprimer dans l'algèbre de points en considérant les instants de début et de fin de chaque intervalle [Ghallab 89.2]. Par exemple, si on veut exprimer que (I recouvre J) avec $I=[I_1,I_2]$ et $J=[J_1,J_2]$, on peut écrire : ($I_1<J_1$ et $J_1<I_2$ et $I_2<J_2$).

Mais toutes les relations de l'algèbre d'intervalles ne sont pas traduisibles sous cette forme. Prenons le cas de (I avant ou après J) qui s'écrit aussi [(($I_2<J_1$) ou ($J_2<I_1$)) et ($I_1<I_2$) et ($J_1<J_2$)]. Cette dernière expression n'appartient pas à l'algèbre d'instants puisqu'elle comporte une contrainte ($I_2<J_1$) ou ($J_2<I_1$) ; cette disjonction ne porte pas sur les mêmes variables et n'est pas représentable dans l'algèbre d'instants. On peut démontrer qu'il existe un sous ensemble de l'algèbre d'intervalles qui est équivalent à l'algèbre d'instants, l'algèbre d'intervalles restreinte. Elle comporte 187 relations dont les relations équivalentes aux primitives d'Allen.

En pratique, lorsque l'on travaille sur l'algèbre d'intervalles on se restreint souvent aux primitives d'Allen et on peut donc représenter la plupart des problèmes de façon équivalente dans l'algèbre d'instants ou dans l'algèbre d'intervalles. On choisit souvent l'algèbre d'instants, plus commode à manipuler. Si l'on utilise toutes ses relations, l'algèbre d'intervalles est plus riche que l'algèbre d'instants d'un point de vue expressivité mais les algorithmes sont plus complexes à traiter et moins efficaces.

II.1.2 Représentations numériques

A) Contraintes élémentaires

L'élément de base dans les représentations numériques est le point [Fontaine 93], [Dousson 94.2], [Dechter 91]. Chaque nœud représente une date, il peut modéliser un événement ponctuel, le début ou la fin d'un événement ayant une durée. Dans le cas le plus général, les relations temporelles sont des disjonctions d'intervalles numériques quelconques. Une relation élémentaire [a,b[d'un nœud X_1 vers X_2 signifie que le délai entre X_1 et X_2 est compris entre a et b : $a \leq X_2 - X_1 < b$. Mais on se limite la plupart du temps à des intervalles tous fermés, ce qui évite d'avoir à calculer si les bornes des intervalles sont ouvertes ou fermées.

Voici un exemple de situation décrite en langage naturel et sa représentation sous forme de graphe [Dechter 91]. : *"John va à son travail soit en voiture, auquel cas cela doit lui prendre de 30 à 40 minutes inclus, soit en bus, auquel cas cela doit lui prendre au moins 60 minutes. Fred va à son travail en voiture (de 20 à 30 mn), ou en bus (de 40 à 50 mn). John quitte généralement sa maison entre 7h10 et 7h20, et Fred arrive au travail entre 8h et 8h10. Enfin, John arrive normalement 10 à 20 minutes après que Fred a quitté sa maison."*

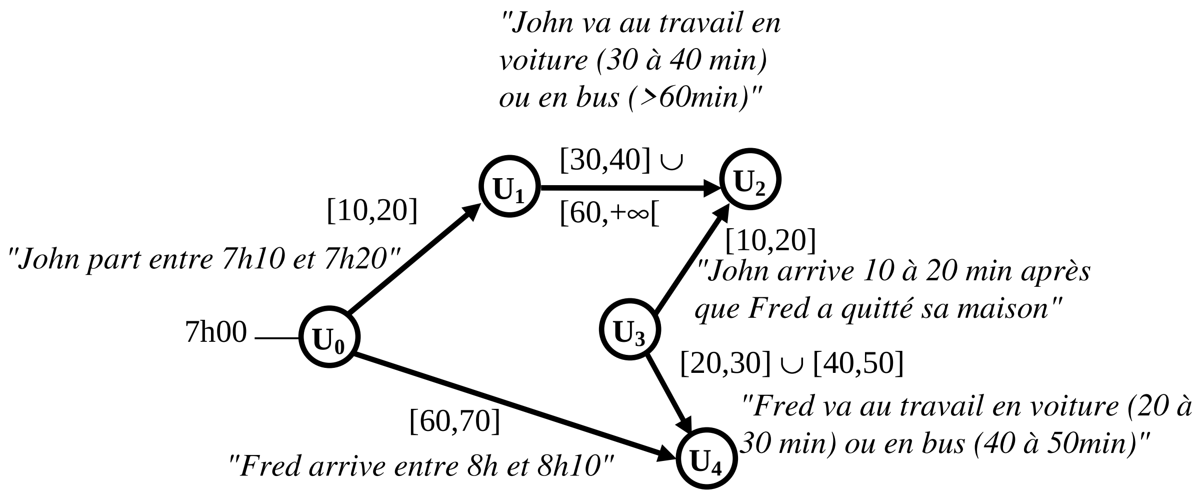


Fig. 8 : représentation numérique

On peut représenter une date absolue par une relation avec une date de référence. Une contrainte entre deux points peut modéliser soit un délai entre deux événements, soit la durée d'une activité. La relation]-∞,+∞[signifie l'ignorance totale sur la relation. On peut aussi traduire certaines relations symboliques sous cette forme. La relation "avant" se traduit par l'intervalle [0,+∞[par exemple. On parle de graphe disjonctif lorsque au moins une des relations du graphe est effectivement disjonctive, et de graphe simple sinon.

B) Expression de contraintes symboliques

Les approches symboliques fournissent une représentation plus souple que les approches numériques. Dans certaines applications, une représentation numérique serait trop rigide. On peut en effet travailler sur des variables temporelles qui sont quantifiées mais pour lesquelles on est incapable de fournir une interprétation numérique. Mais la représentation numérique est inévitable dès que les données contiennent des indications quantitatives sur les durées et les dates. On perdrait des informations en se limitant à l'aspect symbolique. De plus on peut exprimer certaines notions symboliques sous forme métrique comme on va le voir. Une autre façon de faire est de combiner numérique et symbolique (cf. II.1.3).

Dechter [Dechter 91] distingue les TCSP (Temporal Constraint Satisfaction Problems) et les STP (Single Temporal Problems) où les contraintes temporelles sont des intervalles fermés respectivement disjonctifs et non disjonctifs (ou simples). Les TCSP ont un pouvoir représentatif plus riche mais sont aussi beaucoup plus complexes à manipuler. Toutes les relations de l'algèbre d'instants sont représentables par les TCSP, alors que les STP ne permettent pas de représenter la relation "≠" (< ou >) [Fontaine 93].

On a vu l'équivalence entre l'algèbre d'intervalles restreinte et l'algèbre d'instants ; il en découle que l'on peut représenter les relations de l'algèbre d'intervalles par un TCSP. L'algèbre d'intervalles restreinte comporte entre autres toutes les primitives d'Allen. Par exemple, la relation "I recouvre J" avec $I=[I_1,I_2]$ et $J=[J_1,J_2]$ se modélise par :

$$\text{I1} \xrightarrow{]0,+\infty[} \text{J1} \xrightarrow{]0,+\infty[} \text{J2} \xrightarrow{]0,+\infty[} \text{I2}$$

Fig. 9 : représentation numérique de "I recouvre J"

Certaines relations de l'algèbre d'intervalles restreinte ne peuvent plus être représentées par des intervalles simples, elles ne sont donc plus exprimables par des STP. C'est le cas de toutes les relations qui se traduisent dans l'algèbre d'instants à l'aide de la primitive "≠". Les primitives d'Allen restent traduisibles sous la forme de STP.

II.1.3 Représentations mixtes

L'idéal pour être capable de décrire n'importe quelle situation est de pouvoir mélanger la représentation symbolique avec la représentation numérique. Ceci a déjà été envisagé sous deux formes différentes [Kautz 91.1], [Meiri 91].

Kautz et Ladkin [Kautz 91.1] maintiennent en permanence deux graphes distincts. L'un est un graphe du type Allen où l'on peut exprimer toutes les relations symboliques entre intervalles. L'autre est un graphe du type STP à la différence près que l'on peut avoir des intervalles fermés, ouverts, semi-ouverts ou semi-fermés. Ce dernier permet de compléter les relations exprimées par le réseau d'Allen en ajoutant des précisions sur les dates absolues de début ou de fin d'un intervalle, sur les durées des intervalles ou sur les délais entre eux.

Pour tout intervalle I du graphe d'Allen, on introduit dans le STP les deux instants I_L et I_R de début et de fin de l'intervalle avec la relation obligatoire $I_L<I_R$ et toutes les relations supplémentaires inexprimables dans l'algèbre d'intervalles. Il y a bien sûr des implications entre les deux graphes. Par exemple si I(b)J alors on peut déduire $I_R-J_L<0$ et réciproquement. D'autres relations faisant intervenir plusieurs contraintes numériques sont plus difficiles à traduire : si $3<(I_R-I_L)<\infty$ et $-\infty<(J_R-J_L)<2$ alors il faut inférer que I ne peut pas être pendant J puisqu'il dure plus longtemps. Deux algorithmes permettent de passer d'un graphe à l'autre en perdant un minimum d'information. Lors de tout raisonnement, on effectuera à chaque modification d'un graphe une traduction dans son graphe complémentaire.

Chez Meiri [Meiri 91], toute l'information est regroupée dans un seul graphe dont les nœuds peuvent être des instants ou des intervalles. On retrouve les relations symboliques de l'algèbre d'intervalles ou de l'algèbre d'instants ainsi que les relations numériques entre points du type TCSP avec des intervalles fermés, ouverts, semi-ouverts ou semi-fermés. On trouve en plus des relations qualitatives entre un point et un intervalle ou entre un intervalle et un point.

On forme avec toutes les relations symboliques une algèbre qualitative comportant les 2^{13} relations de l'algèbre d'intervalles, les 2^3 relations de l'algèbre de points, les 2^5 relations intervalle/point et enfin les 2^5 relations point/intervalle. Pour tout intervalle i, on introduit les contraintes (I_L débute I) et (I_R finit I).

Entre deux points, on peut avoir soit une relation symbolique, soit une relation numérique. On maintient en permanence les deux types de relations en établissant une équivalence entre l'algèbre d'instants et les relations métriques entre instants. D'un point de vue conceptuel, cette représentation est plus satisfaisante que celle de Kautz et Ladkin puisqu'elle regroupe toutes les données en un seul graphe et qu'elle permet des relations symboliques supplémentaires point/intervalle, intervalle/point qui chez Kautz auraient été traduites par des contraintes numériques, intuitivement moins parlantes.

II.1.4 Représentations floues

A) Eléments temporels de base

La plupart des représentations temporelles floues s'inspirent de [Dubois 89] ou [Barro 94]. Chez Dubois, le temps est modélisé par la droite des réels alors que chez Barro c'est la droite des entiers naturels. Pour ce dernier, la discrétisation du temps se fait suivant la précision requise (granularité) pour les informations temporelles à modéliser : si toutes les données sont connues à une minute près par exemple, une unité sur l'axe du temps représentera une minute. Dans de nombreux problèmes, cette modélisation fournit une précision suffisante pour les calculs effectués et permet une implémentation plus simple.

a) dates

Les dates sont décrites par des ensembles flous μ_d dans ces deux formalismes. $\mu_d(x)$ est le degré d'appartenance de l'instant x à la date d. d est l'ensemble de tous les instants qui représentent la date. On peut aussi interpréter l'ensemble flou μ_d comme une distribution de possibilités π_d, auquel cas $\pi_d(x)$ est le degré de possibilité pour x d'être réellement cette date d. Pour une date donnée, il est plus ou moins possible que cette date corresponde précisément à l'instant que l'on cherche à représenter. Suivant la nature de cet instant, la distribution de possibilités peut prendre plusieurs formes.

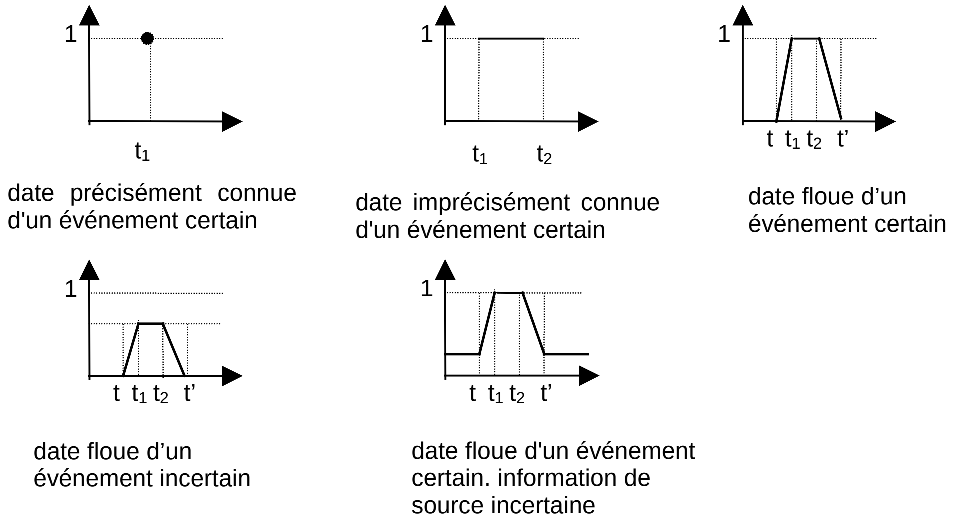


Fig. 10 : différentes représentations floues de dates

Les distributions de possibilités sont impérativement convexes (ou unimodales), et chez Barro elles doivent en plus être normalisées, ce qui exclut de représenter des événements dont l'occurrence est incertaine comme chez Dubois.

b) durées

Les durées sont elles aussi représentées par des ensembles flous. Si on a deux instants flous t_1 et t_2, la durée floue qui les sépare peut être obtenue en appliquant le principe d'extension pour calculer la différence floue $t_2 \ominus t_1$. La distribution de possibilités de la durée entre t_1 et t_2, $\pi_{\delta_{21}}(l)$, est donnée par :

$$\Pi_{\delta_{21}}(l) = \max_{\substack{\tau_1 \in S_{t_1} \\ \tau_2 \in S_{t_2} \\ \tau_2 - \tau_1 = l}} (\min(\pi_{t_1}(\tau_1), \pi_{t_2}(\tau_2)))$$ où S_{t_1} et S_{t_2} sont les supports de π_{t_1} et π_{t_2}

La distance entre deux instants flous convexes et normalisés est également convexe et normalisée. Si t_1 et t_2 sont deux distributions trapézoïdales $(\alpha_i,\beta_i,\gamma_i,\delta_i)$, alors $t_2 \ominus t_1$ a aussi une distribution trapézoïdale de la forme $(\alpha_2-\delta_1,\beta_2-\gamma_1,\gamma_2-\beta_1,\delta_2-\alpha_1)$. Une durée $t_2 \ominus t_1$ négative (resp. positive) correspond à t_2 avant (resp. après) t_1.

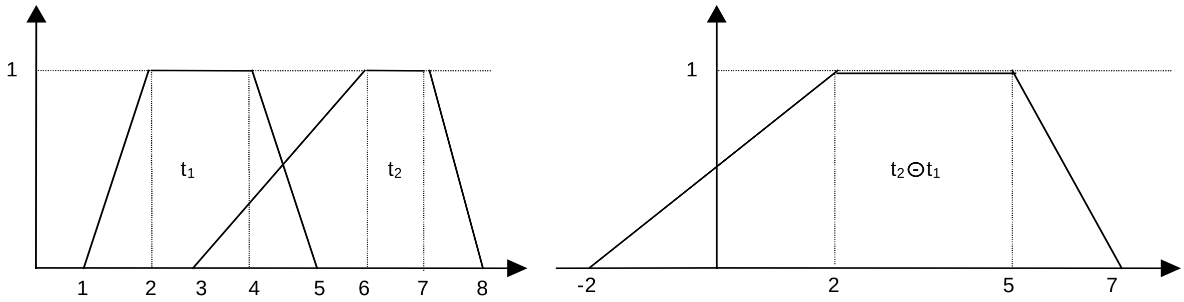


Fig. 11 : calcul d'une durée à partir de dates floues

c) intervalles

Les points de vue commencent à diverger légèrement lorsqu’il s’agit de représenter des intervalles de temps continus. Pour Steunou [Steunou 92] et Chen [Chen 93], un intervalle est simplement un couple d’instants flous représentant le début et la fin de l’intervalle. Tous les intervalles sont traités en gardant cette forme.

Pour Dubois et Prade [Dubois 89], un intervalle flou est aussi formé à partir de ses instants flous de début a et de fin b, mais on peut lui donner une représentation spécifique en appliquant le principe d'extension.

$$\mu_{[A,B]}(t) = \sup_{s \le t \le s'} \min(\mu_A(s), \mu_B(s'))$$
$$\mu_{]A,B[}(t) = \sup_{s < t < s'} \min(\mu_A(s), \mu_B(s'))$$

Notons que le support de $]A,B[$ est inclus dans le noyau de [A,B]. [A,B] correspond à la possibilité de se situer entre A et B alors que $]A,B[$ correspond à la nécessité du même fait. On a bien un degré de possibilité de 1 lorsque le degré de nécessité est strictement positif, ce qui est indispensable en théorie des possibilités.

Marin [Marin 94] exprime un point de vue un peu différent. Un intervalle est non plus un couple mais un triplet de données floues contenant toujours les dates de début A et de fin B de l’intervalle mais avec en plus sa durée floue D positive. On déduit de (A,B,D) une représentation minimale (A’,B’,D’) de l’intervalle.

A' contient toutes les dates de début de l'intervalle compatibles avec une durée D et une date de fin dans B. De même pour B' et D' qui contiennent les dates de fin et les durées possibles.

$A'=A\cap(B\ominus D)$ $B'=B\cap(A\oplus D)$ $D'=D\cap(B\ominus A)$

B) Contraintes temporelles floues

La représentation sous forme de graphes temporels s'établit sur une base ponctuelle : les nœuds du graphe sont des dates. Un intervalle sera donc figuré par une décomposition en sa date de début et sa date de fin. Chez Barro [Barro 94], les arcs du graphe sont étiquetés par des durées floues ; elles modélisent des contraintes de distances entre dates. Dubois [Dubois 89] est plus général et les arcs sont des relations floues. Enfin chez Chen [Chen 93] les relations sont les relations symboliques classiques de l'algèbre d'instants mais associées à un couple (degré de possibilité, degré de nécessité).

a) Relations floues

La relation < peut par exemple s'exprimer par la relation floue $\pi_<(s,t)=1$ si s<t et 0 sinon. Pour la précédence stricte comme "très nettement avant", une nuance peut être donnée par une relation floue $\pi_R(s,t)=\min(1,(t-s)/C)$ si s<t et 0 sinon, C étant une constante choisie de façon adéquate pour représenter le terme "très nettement". Les relations <, >, ≤ ou ≥ ainsi obtenues correspondent aux définitions classiques de possiblement/nécessairement inférieur/supérieur exprimées comme suit dans le contexte possibiliste [Dutta 91.1]:

- l'intervalle flou $[A,+\infty[$ des dates possiblement après A : $\mu_{[A,+\infty[}(t)=\sup_{s\le t}(\pi_A(s))$
- l'intervalle flou $]-\infty,A]$ des dates possiblement avant A : $\mu_{]-\infty,A]}(t)=\sup_{s\ge t}(\pi_A(s))$
- l'intervalle flou $]A,+\infty[$ des dates nécessairement après A : $\mu_{]A,+\infty[}(t)=\inf_{s\ge t}(1-\pi_A(s))$
- l'intervalle flou $]-\infty,A[$ des dates nécessairement avant A : $\mu_{]-\infty,A[}(t)=\inf_{s\le t}(1-\pi_A(s))$

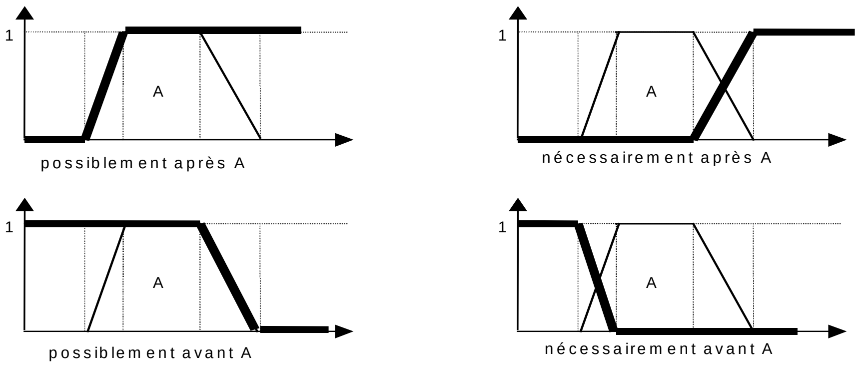


Fig. 12 : relation d'ordre sur les intervalles flous

La distribution de possibilités d'un couple de dates (a,b) relié par une relation R et définies chacune par π_a et π_b est :

- $\pi_{(a,b)}(s,t)=\min(\pi_a(s), \pi_b(t), \mu_R(s,t))$

Et la combinaison de la date a avec la relation R donne pour b :

- $\pi_b(t)=\sup_s \min(\pi_a(s), \mu_R(s,t))$

On peut traduire toutes les relations symboliques de l'algèbre d'intervalles en combinant ces relations [Dutta 91.1]. Il est cependant plus difficile de manipuler des relations floues que des durées floues, même si elles ont un plus grand pouvoir d'expression. On essaiera la plupart du temps de se ramener à des contraintes de durées floues comme chez Barro [Barro 94], qui, comme nous allons le voir, permettent déjà de représenter un bon nombre de notions symboliques ou quantitatives.

b) Durées floues

Chez Barro un arc (i,j) porte la contrainte de durée P(i,j) de fonction d'appartenance $\mu_{P(i,j)}$. Pour exprimer une date absolue a on établit une contrainte P(0,a) entre un instant de référence de date 0 et un nœud a avec $\mu_{P(0,a)}(x)=\mu_a(x)$. Pour une date b relative à une date a on a un arc de contrainte P(a,b) qui exprime la durée d entre a et b. Pour un intervalle [a,b] de durée d on a 3 contraintes P(0,a), P(0,b) et P(a,b) avec :

- $\mu_{P(0,a)}(x)=\mu_a(x)$
- $\mu_{P(0,b)}(x)=\mu_b(x)$
- $\mu_{P(a,b)}(x)=\mu_d(x)$

On peut ainsi exprimer des dates absolues numériques ou symboliques comme "à 10h00", "hier", "2 heures avant", des relations numériques entre dates, entre intervalles, entre une date et un intervalle ou l'inverse. On peut aussi modéliser toutes les primitives classiques des relations symboliques entre ces mêmes entités (avant, après, pendant …). Les relations avec des intervalles s'obtiennent en les décomposant en relations avec les dates de début et de fin de l'intervalle.

L'utilisation de durées plutôt que de relations floues ne permet pas de retrouver les relations classiques de "nécessairement avant" et "nécessairement après". Elles ne sont en effet pas exprimables par des durées, on peut seulement s'en approcher en définissant une durée "nécessairement positive" (Fig. 13).

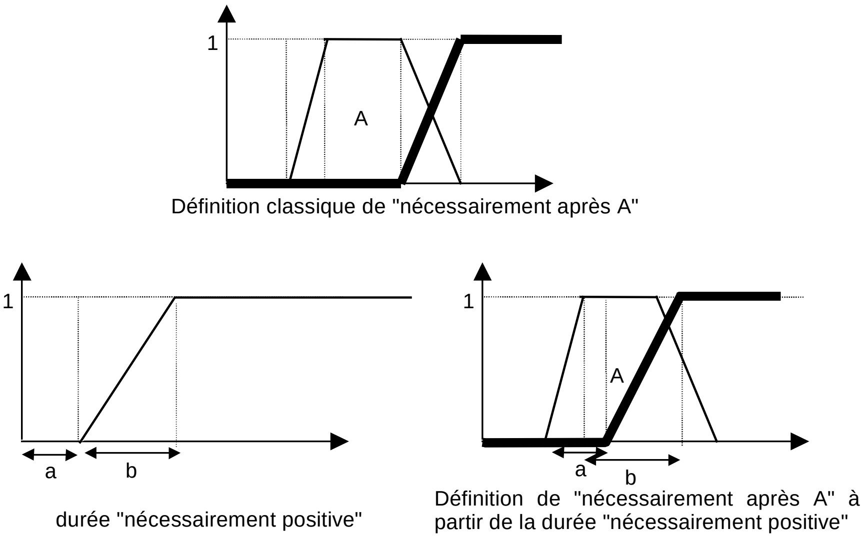


Fig. 13 : différentes définitions de "nécessairement après A"

Les modificateurs linguistiques comme "très", "environ" sont représentables par des nuances sur les fonctions d'appartenance obtenues par des translations, des homothéties…

Il est toujours possible de compléter le graphe par des "contraintes unité" de distribution à valeur 1 sur tout l'univers. Comme dans les autres formalismes présentés, on ne peut pas comparer des durées.

c) Relations possibilistes

Chen adopte lui une autre modélisation des contraintes [Chen 93]. Les relations utilisées sont les primitives de l'algèbre d'instants <,> et =. Mais on leur associe en plus un degré de possibilité et un degré de nécessité, ce qui permet de témoigner une incertitude sur la relation. La possibilité d'avoir une relation (R) entre deux dates floues t_1 et t_2 de distributions de possibilités π_{t_1} et π_{t_2} est :

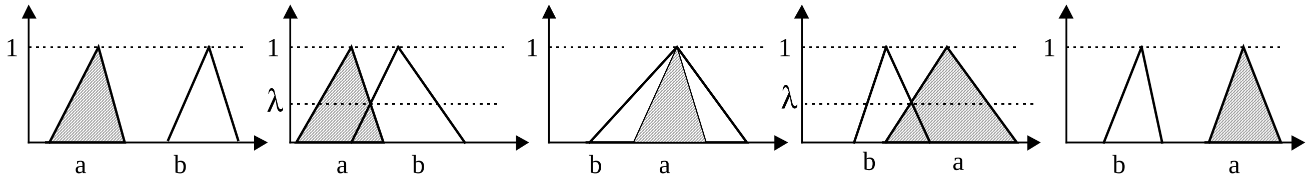
$$\Pi((R)(t_1,t_2)) = \max_{\substack{t_1 \in S_{t_1} \\ t_2 \in S_{t_2} \\ t_1 (R) t_2}} (\min[\pi_{t_1}(t_1), \pi_{t_2}(t_2)]) \qquad \textbf{(1)}$$

La nécessité est :

$$N((R)(t_1,t_2)) = 1 - \max_{\substack{t_1 \in S_{t_1} \\ t_2 \in S_{t_2} \\ t_1 (\neg R) t_2}} (\min[\pi_{t_1}(t_1), \pi_{t_2}(t_2)]) \qquad \textbf{(2)}$$

où $(\neg R)$ est la disjonction des comparateurs logiques autres que (R).

Avec des distributions de possibilités triangulaires et en considérant les relations <,> et = on trouve cinq configurations possibles :



	Π	N	Π	N	Π	N	Π	N	Π	N
a<b	1	1	1	1- λ	1^-	0	λ^-	0	0	0
a>b	0	0	λ^-	0	1^-	0	1	1- λ	1	1
a=b	0	0	λ	0	1	0^+	λ	0	0	0

Fig. 14 : configurations sur l'algèbre d'instants

Les contraintes faisant intervenir des intervalles sont calculées à partir des contraintes sur les instants. Pour établir une relation entre une date et un intervalle on la compare aux deux bornes de cet intervalle. Pour comparer deux intervalles, on retrouve les 13 relations d'Allen. Par exemple, pour deux intervalles i et j définis par leurs débuts respectifs t_{id} et t_{jd} et leurs fins respectives t_{if} et t_{jf}, la contrainte "i pendant j" est évaluée par :

$$\pi(\text{i pendant j})=\min(\pi(t_{id}>t_{jd}),\pi(t_{if}<t_{jf})) \text{ et } N(\text{i pendant j})=\min(N(t_{id}>t_{jd}),N(t_{if}<t_{jf}))$$

C) Construction des ensembles flous

Les fonctions d'appartenance peuvent être construites de façon automatique en utilisant une grammaire [Barro 94] ou de façon plus manuelle d'après le recueil d'expertise. Dans une grammaire, on associe à chaque terme du langage un ensemble flou ou une modification sur l'ensemble flou d'un terme adjacent, et pour traduire une expression complète, il suffit de faire la somme des effets des mots qui la composent.

Voici un exemple tiré de [Barro 94]. On considère l'expression "environ un mois avant le 15 Février 1993".

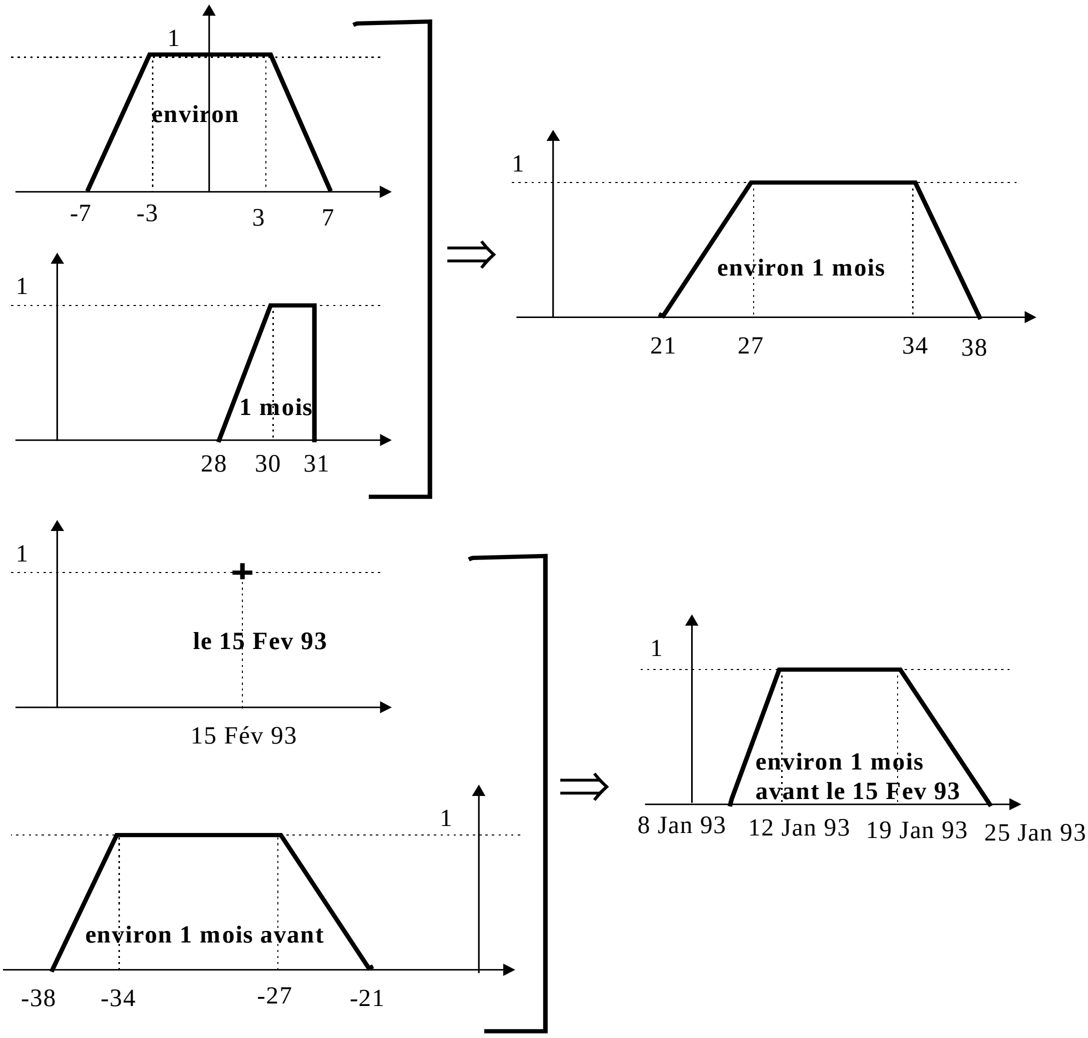


Fig. 15 : construction de "environ un mois avant le 15 Février"

"Un mois" est un ensemble flou de support 28 à 31 jours et de noyau 30 à 31 jours. On lui applique le modificateur "environ" en lui ajoutant l'ensemble flou qui représente cette notion. C'est un ensemble flou symétrique autour de 0 dont le rôle est d'élargir l'imprécision du nombre flou auquel on le somme. La notion "avant" porte sur "environ un mois" et "environ un mois avant" est obtenu en prenant l'opposé de l'ensemble flou représentant "environ un mois". Le 15 Février 1993 est une date précise et l'expression finale est la somme de l'ensemble flou "environ un mois avant" et de l'ensemble classique "15 Février 1993".

Pour une construction des fonctions d'appartenance par expertise, il est assez facile d'obtenir le support et le noyau en demandant à l'expert l'intervalle de valeurs qui correspond tout à fait à la notion que l'on veut représenter et les valeurs limites au delà desquelles cette notion ne s'applique plus. On obtient ainsi des fonctions d'appartenance trapézoïdales.

D) Avantages de la représentation

Finalement, le formalisme flou apparaît aussi riche que les représentations mixtes, avec en plus la possibilité de tenir compte d'imprécisions et d'incertitudes. Un avantage supplémentaire est de représenter toutes les contraintes sous une forme unique, l'ensemble flou. On ne distingue plus le symbolique et le numérique à part dans le langage de description. Dès que l'on a construit les fonctions d'appartenance, ce qui se fait hors ligne et donc sans contrainte temporelle, le traitement des données est totalement unifié et ne nécessite pas de distinction entre le raisonnement sur le numérique et celui sur le symbolique.

II.2 Raisonnements temporels

Le choix d'une modélisation est généralement lié à l'utilisation que l'on veut en faire. Suivant la modélisation retenue, on pourra effectuer certains types de raisonnements et pas d'autres. Nous allons décrire les raisonnements les plus couramment utilisés et leurs modélisations.

II.2.1 Approche du type CSP

Les approches de type CSP (**C**onstraints **S**atisfaction **P**roblems) sont de loin les plus utilisées et sont assez générales pour être applicables sur de nombreuses modélisations [Schwalb 98]. On ne présente pas ici leur application sur des modélisations floues qui donne lieu à une extension des CSP, les FCSP (**F**uzzy **C**onstraints **S**atisfaction **P**roblems) (cf. III.1 p.73).

Dans les CSP, un problème est représenté par un graphe où les nœuds sont les éléments contraints (des dates d'événements dans notre cas) et les arcs orientés sont étiquetés par les contraintes. Les techniques utilisées par les problèmes de satisfaction de contraintes sont fondées sur les 3 opérations élémentaires entre contraintes : inverse, intersection et composition. Ces opérations prennent différentes formes suivant le type de contraintes, c'est - à-dire suivant l'algèbre sur laquelle on se place.

A) Opérations élémentaires

a) Inverse

Lorsque l'on a une contrainte C(i,j) entre un nœud i et un nœud j, on peut en déduire une contrainte inverse C(j,i) entre j et i. Pour toutes les contraintes symboliques, l'inverse des primitives est fourni par des tableaux (m inverse de m_i, > inverse de <…). Si C(i,j) est une disjonction de primitives, C(j,i) est la disjonction des inverses des primitives. L'inverse d'une contrainte numérique $C(i,j)=[a,b]\cup]c,d]\cup\ldots\cup[y,z[$ est $C(j,i)=[-b,-a]\cup[-d,-c[\cup\ldots\cup]-z,-y]$.

b) Intersection

L'intersection est aussi une opération très fréquente sur les contraintes. Si l'on a deux contraintes sur un même arc, la contrainte résultante est l'intersection des deux. Pour des contraintes symboliques, l'intersection de deux contraintes primitives r_1 et r_2 est vide si $r_1 \neq r_2$ et elle vaut $r_1=r_2$ sinon. L'intersection de deux contraintes symboliques disjonctives est la disjonction des primitives communes qui les composent. (b ou m ou f)$\cap$(s ou o_i ou m)=(m) par exemple. Pour des contraintes numériques classiques : $[a,b]\cap[c,d]=[\max(a,c),\min(b,d)]$. Pour des intervalles non fermés, on raisonne ainsi : $]a,b]\cap[c,d]=]\max(a,c),\min(b,d)]$ si $\max(a,c)=a$ et $[\max(a,c),\min(b,d)]$ si $\max(a,c)=c$. Pour des disjonctions d'intervalles (ou chaînes d'intervalles), le résultat est la disjonction des intersections des intervalles deux à deux :

$$([a_1,b_1]\cup[a_2,b_2])\cap([c_1,d_1]\cup[c_2,d_2])=$$
$$([a_1,b_1]\cap[c_1,d_1])\cup([a_1,b_1]\cap[c_2,d_2])\cup([a_2,b_2]\cap[c_1,d_1])\cup([a_2,b_2]\cap[c_2,d_2])$$

c) Composition

Enfin, la composition est l'opération de base pour la propagation. Si l'on a une contrainte C(i,j) entre i et j et une autre C(j,k) entre j et k alors on peut déduire la contrainte C(i,k) entre i et k. C(i,k) est la composition des deux premières contraintes. De même que pour les inverses, les composées de primitives symboliques sont fournies par des tableaux [Allen 81], [Mc Dermott 82]. Pour des disjonctions, on effectue la composition des primitives deux à deux et le résultat est la disjonction de ces composées.

La composition de contraintes numériques sous forme d'intervalles se ramène à la somme d'intervalles : $[a,b]_\circ[c,d]=[a+c,b+d]$. Pour des intervalles non fermés et les disjonctions, c'est le même principe que pour l'intersection en remplaçant l'intersection de deux intervalles élémentaires par leur composition :

$$]a,b]_\circ[c,d]=]a+c,b+d] \text{ et}$$
$$([a_1,b_1]\cup[a_2,b_2])_\circ([c_1,d_1]\cup[c_2,d_2])=$$
$$[a_1+c_1,b_1+d_1]\cup[a_1+c_2,b_1+d_2]\cup[a_2+c_1,b_2+d_1]\cup[a_2+c_2,b_2+d_2]$$

B) Notions de base

a) Consistance et inconsistance

Les opérations élémentaires décrites ci-dessus, associées à certaines notions basiques, permettent de réaliser les raisonnements fondamentaux sur ce type de structures. Les notions primordiales sont l'inconsistance, la consistance locale et la minimalité. On dit qu'un problème est inconsistant lorsqu'il n'existe aucune solution vérifiant toutes ses contraintes. Si une contrainte du graphe devient vide, alors le problème est nécessairement inconsistant. Si au contraire il existe au moins une solution, alors on dit que le problème est consistant.

Mais il n'est pas toujours facile de trouver une solution, et il existe d'autres moyens de caractériser la consistance [Fargier 94] qui peuvent faciliter la recherche d'une solution. Des critères de consistance locale (k-consistance, consistance de chemins) sont souvent beaucoup plus faciles à caractériser et permettent d'éliminer certaines valeurs des domaines des variables. Il y a consistance locale lorsqu'il existe une solution locale, c'est-à-dire une solution qui vérifie les contraintes entre certains nœuds du graphe.

b) k-consistance

On distingue en fait plusieurs niveaux de consistance locale. Le plus faible est la 2-consistance. Il y a 2-consistance lorsque pour tout élément vérifiant la contrainte unaire sur un nœud i quelconque il existe un élément vérifiant la contrainte unaire sur n'importe quel autre nœud j et tel que la contrainte binaire entre i et j soit vérifiée. Il y a en plus arc-consistance s'il existe au moins un élément vérifiant la contrainte unaire de chaque nœud i.

La k-consistance généralise la notion de 2-consistance. Un graphe est k-consistant si toute instanciation localement consistante de k-1 variables peut s'étendre à une instanciation localement consistante de k variables. La k-consistance forte est la m-consistance pour tout m / m≤k. La n-consistance forte pour un graphe comportant n nœuds implique la consistance globale.

c) Consistance de chemins

La propriété de consistance de chemins est plus faible que la n-consistance forte mais est beaucoup plus utilisée car plus simple à mettre en œuvre. Soit un couple (i,j) de nœuds pour lequel il existe un couple solution vérifiant les contraintes unaires et binaires sur i et j. Alors il y a consistance de chemins si pour n'importe quel chemin reliant i à j, on peut étendre la solution à une solution vérifiant toutes les contraintes unaires et binaires du chemin. Les autres contraintes du chemin n'induisent pas de contrainte plus restreinte entre i et j.

On dit qu'un graphe est complet lorsque tout couple de nœuds est lié par une contrainte. Dans tous les cas de graphes vus ici, on peut obtenir un graphe complet équivalent au graphe initial (ayant exactement les mêmes solutions). La particularité d'un graphe complet est que la consistance de chemins est équivalente à la consistance des chemins de longueur 2 (3-consistance) : c'est le théorème de Montanari [Montanari 74].

d) Minimalité

La minimalité joue un rôle prépondérant dans la reconnaissance. Cette notion s'appuie sur la définition de l'équivalence. Un graphe est équivalent à un autre s'il comporte les mêmes nœuds et les mêmes solutions. On construit ainsi une relation d'équivalence qui comporte un plus petit élément pour l'inclusion : c'est le réseau minimal, le graphe équivalent qui comporte les contraintes les plus restrictives possibles [Dechter 91], [Fargier 94]. Toute solution partielle vérifiant les contraintes du graphe minimal peut être étendue en une solution globale. Si le graphe minimal ne comporte pas de contrainte vide, alors le problème est consistant.

Un deuxième théorème de Montanari ajoute que si un graphe est chemins-consistant et que la composition des contraintes est distributive sur l'intersection, alors le graphe est minimal. Davis généralise ces deux théorèmes au cas de contraintes numériques : si les contraintes sont non disjonctives alors le filtrage par 3-consistance aboutit à la minimalité.

e) Filtrage

Le principe du raisonnement sur des graphes de contraintes est le suivant : on filtre les graphes par propagation des contraintes selon un critère de consistance locale, ce qui veut dire que l'on construit par des mécanismes de propagation un graphe équivalent avec des contraintes plus explicites qui doivent impérativement vérifier le critère de consistance locale choisi. Si le filtrage est optimal, on obtient le graphe minimal.

Celui-ci permet de répondre à la question de l'existence d'une solution. Il facilite aussi une recherche incrémentale de solution. On instancie une première variable dans son domaine de valeurs, ce qui fixe de nouvelles contraintes, puis on recherche le nouveau graphe minimal et ainsi de suite jusqu'à l'instanciation d'une solution complète. Nous allons voir maintenant les algorithmes qui permettent d'atteindre ce résultat pour différents graphes temporels.

C) Algorithmes

a) Algorithmes de 3-consistance

La plupart des algorithmes du type CSP réalise un filtrage par consistance des chemins de longueur 2 (3-consistance). Ils transforment un graphe en un graphe équivalent mais qui, par construction, est 3-consistant. Le principe général de ces algorithmes consiste à propager les contraintes entre trois nœuds pour en déduire de nouvelles contraintes plus précises. L'algorithme se poursuit tant que l'on peut affiner des contraintes, et s'il s'arrête on a un graphe 3-consistant. Ces algorithmes connaissent plusieurs variantes, ils sont connus sous les noms d'algorithmes de Waltz, de Montanari ou de Mackworth.

Les deux algorithmes les plus fréquemment cités à propos des CSP classiques pour le calcul du graphe minimal sont l'algorithme de Montanari (PC1) et l'algorithme de Mackworth (PC2). Leur principe est le même mais PC2 est plus efficace. Il s'agit de propager toutes les contraintes sur tous les chemins possibles de longueur 3 jusqu'à ce qu'il n'y ait plus aucune modification. C_{ij} décrit la contrainte entre le nœud i et le nœud j.

Algorithme PC1 :

- 1. Tant que le graphe est modifié faire
- 2. de k=1 à n
- 3. de i,j =1 à n
- 4. $C_{ij}=C_{ij}\cap(C_{ik}\circ C_{kj})$ (intersection et composition définies selon le type de contraintes)
- 5. Si $C_{ij}=\varnothing$ alors le graphe est inconsistant. Arrêt

Algorithme PC2 :

- 1. $Q=\{(i,j,k)/\ i\leq j$ et non$(i=j=k)\}$
- 2. Tant que Q non vide faire
- 3. Choisir et supprimer un triplet (i,j,k) de Q
- 4. $C_{ij}=C_{ij}\cap(C_{ik}\circ C_{kj})$ (intersection et composition définies selon le type de contraintes)
- 5. Si $C_{ij}=\varnothing$ alors le graphe est inconsistant. Arrêt
- 6. Si C_{ij} a été modifiée alors
- 7. chemins_liés$(i,j,k)=\{(i,m,j)/\ i\leq m,\ m\neq j\}\cup\{(m,j,i)/\ m\leq j,\ m\neq i\}\cup\{(j,m,i)/\ j<m\}\cup\{(m,i,j)/\ m<i\}$
- 8. $Q=Q\cup$chemins_liés(i,j,k)

A la différence de PC1, PC2 met à jour la liste des chemins qui doivent encore être propagés. Il est en théorie plus efficace, mais en pratique le gain apporté par la gestion des chemins à propager ne compense pas toujours la perte de temps liée à la mise à jour de cette liste.

Le théorème de Montanari assure d'autre part l'équivalence entre consistance des chemins de longueur 2 et consistance de chemins pour le cas de graphes complets. Ce type d'algorithmes permet donc d'obtenir des graphes chemins-consistants à partir des graphes complétés. Ces algorithmes ne détectent pas forcément l'inconsistance et n'entraînent pas la minimalité dans le cas général. Nous allons voir suivant le type de graphe temporel étudié la nature du résultat et la complexité de l'algorithme. Une question importante est aussi de savoir si l'algorithme se termine.

b) Cas symbolique

Pour les CSP avec un nombre fini de contraintes possibles, comme c'est le cas dans l'algèbre d'instants (PA) ou dans l'algèbre d'intervalles (IA), l'algorithme de 3-consistance est en $O(n^2)$ et la recherche du graphe minimal est un problème NP-complet. Il est complet pour PA mais ne fournit pas le graphe minimal et est incomplet pour IA.

Mais on peut simplifier ces problèmes dans certains cas. Si l'on considère l'algèbre de points privée de la relation ≠ (PA*) ou la sous-classe d'IA correspondante ou encore les primitives d'IA, les algorithmes de consistance de chemins trouvent le graphe minimal [Van Beek 92].

Une autre sous-classe de l'algèbre d'intervalles intéressante a été introduite par Nebel et Bürckert [Nebel 95], la sous-classe des relations ORD-Horn. Cette sous-classe comporte 868 relations. Elle est composée de toutes les relations pouvant s'écrire à l'aide des relations ≠, ≤ et = entre les points de début et de fin des intervalles et comportant au moins un littéral positif de la forme $a=b$ ou $a\leq b$. Pour cette sous-classe l'algorithme de consistance de chemins est complet et a une complexité polynômiale. C'est la plus grande sous-classe de l'algèbre d'intervalles pour laquelle on puisse calculer la satisfiabilité en un temps polynômial.

Van Beek transforme l'algorithme de consistance de chemins et obtient pour l'algèbre d'instants complète (PA) le graphe minimal en $O(\max(mn^2,n^3))$ où m est le nombre de contraintes étiquetées par la relation ≠. Pour l'algèbre d'intervalles restreinte S (188 relations), on peut la traduire dans PA et appliquer le même algorithme. Lorsque l'on est confronté à l'algèbre d'intervalles complète, on peut approcher le graphe minimal en se ramenant à des sous-graphes de SA mais dans la pratique on préfère se limiter aux relations de SA et de PA.

c) Cas numérique

La question de savoir si l'algorithme se termine se pose particulièrement dans le cas de CSP numériques pour lesquels les domaines des contraintes sont infinis. Dechter [Dechter 91] distingue parmi les graphes de contraintes temporelles numériques les STP (**S**ingle **T**emporal **P**roblems), pour lesquelles les contraintes sont non disjonctives, et les TCSP (**T**emporal **C**onstraints **S**atisfaction **P**roblems), qui recouvrent toutes les contraintes numériques, disjonctives ou non. Si les domaines sont discrets, l'algorithme se termine en $O(n^3)$ pour les STP et en $O(n^3R^3)$ pour les TCSP, R étant le rang du TCSP. Si l'on travaille avec des entiers naturels et si les contraintes sont de la forme $Ci=[a_{i1},b_{i1}]\cup\ldots\cup[a_{ik},b_{ik}]$ le rang est $R=\max_i(\max_k(b_{ik})-\min_k(a_{ik}))$. Pour des rationnels, c'est le même calcul en multipliant toutes les valeurs par leur PGCD.

Pour les STP en général, le filtrage par 3-consistance donne en fait le graphe minimal. L'algorithme n'aboutit qu'à la chemin-consistance dans le cas de TCSP. Pour trouver le graphe minimal dans ce cas, on peut décomposer le TCSP en une union de STP et le graphe minimal est l'union des graphes minimaux résultant des différents STP. Le TCSP se décompose en un maximum de k^e STP, k étant le nombre maximum d'intervalles par contrainte et e=n(n-1)/2 le nombre d'arcs ; la complexité totale est donc en $O(n^3k^e)$. On essaye donc de limiter au maximum le nombre et la taille des contraintes disjonctives.

d) Cas mixte

Pour les graphes avec une représentation mixte comme dans [Kautz 91.1], il suffit de raisonner séparément sur le STP et sur le graphe d'Allen et de traduire chaque résultat d'un formalisme vers l'autre. Les algorithmes utilisés sont donc les mêmes que précédemment. Pour le STP on obtient le graphe minimal en $O(n^3)$ et pour le graphe d'Allen, on le rend chemins-consistant en $O(n^2)$. A cela il faut ajouter la traduction d'un formalisme à l'autre et les éventuelles modifications supplémentaires qui en découlent. Finalement, l'algorithme global n'est en général pas complet puisque le graphe d'Allen n'est que chemins-consistant.

C'est le même principe dans [Meiri 91] : on raisonne sur les contraintes d'un même type puis on les traduit dans un autre type et ainsi de suite. Le résultat dépend des contraintes initiales : si par exemple on a des contraintes numériques de type STP et des contraintes symboliques de l'algèbre d'instants privée de ≠ ou de son équivalent en intervalles, l'algorithme donne le graphe minimal. Il est nécessaire de bien savoir à quelle catégorie les contraintes appartiennent, ce qui n'est pas toujours facile. Il n'existe pas de critère pour vérifier de l'appartenance à l'algèbre d'intervalles restreinte.

II.2.2 Approche graphe de distances

Les STP peuvent aussi être transformés sous la forme de graphes de distances (d-graphes) et on peut donc leur appliquer les techniques associées [Dechter 91]. Soit une contrainte $T_{ij}=[a_{ij},b_{ij}]$ entre les nœuds X_i et X_j. Sa signification est : $a_{ij} \leq X_j - X_i \leq b_{ij}$ donc $X_j - X_i \leq b_{ij}$ et $X_i - X_j \leq -a_{ij}$. On a donc l'équivalence entre une contrainte de type STP et une relation sur un graphe de distances.

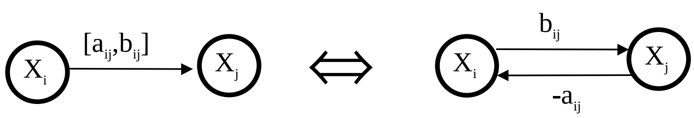


Fig. 16 : transformation d'un STP à un graphe de distances

Tout STP peut donc être transposé en un graphe de distances. Il existe un critère simple pour détecter l'inconsistance dans un graphe de distances : il est inconsistant si et seulement s'il possède un cycle négatif, c'est-à-dire si la somme des distances rencontrées en parcourant un cycle est négative. De plus, on peut utiliser des algorithmes de type "plus court chemin" pour obtenir le graphe des distances minimales. En voici un exemple, l'algorithme de Floyd-Warshall, de complexité en $O(n^3)$, pour un graphe complet où les arcs sont notés d_{ij} :

Algorithme de Floyd-Warshall :

- 1. de i=1 à n faire dii=0
- 2. de k=1 à n faire
- 3. de i,j=1 à n faire
- 4. $d_{ij}=\min(d_{ij},d_{ik}+d_{kj})$

Si on transforme à nouveau le d-graphe ainsi obtenu en un STP, on a alors le STP minimal équivalent au graphe temporel initial. L'algorithme de Floyd-Warshall est donc tout à fait équivalent aux algorithmes de consistance de chemins, et il est de même complexité. Pour un TCSP, on pourra le décomposer en STP de la même façon qu'avec les algorithmes de 3-consistance et la complexité globale sera aussi équivalente. La seule différence entre les deux méthodes tient dans l'utilisation d'un opérateur minimum à la place de l'intersection. Ceci présente un intérêt dans un cas où l'on saurait mieux effectuer une opération plutôt qu'une autre.

II.2.3 IxTeT

Le système IxTeT (**I**nde**x**ed **T**im**e** **T**able) [Ghallab 89.1], [Ghallab 89.2], est dérivé des techniques issues des CSP. On représente aussi le problème par un graphe de contraintes en se plaçant dans l'algèbre d'instants. Mais la structure du graphe est modifiée afin de rendre les opérations de base plus efficaces.

Comme on l'a vu avec les CSP, sur l'algèbre d'intervalles, la vérification de cohérence est un problème NP-complet, et on se limite souvent à une cohérence locale dont la complexité est polynomiale. Dans l'algèbre d'instants, cette opération est beaucoup moins coûteuse, ce qui constitue un de ses principaux intérêts. La 3-consistance est en effet suffisante pour assurer une cohérence globale dans ce cadre [Haton 91].

Dans IxTeT, Ghallab réduit encore la complexité qui apparaît statistiquement linéaire. IxTeT est un Time Map Manager (TMM) complet qui permet un raisonnement non monotone très efficace. Il exécute deux tâches essentielles : pour deux éléments donnés, trouver leur relation temporelle, et effectuer la mise à jour qui consiste à ajouter ou supprimer des éléments de la base tout en maintenant la cohérence. Les objets sont gérés en construisant un arbre optimum avec une indexation pour faciliter la recherche.

Dans un premier temps, on construit un graphe complet où les nœuds sont des instants et les arcs orientés des relations $\leq$ et $\neq$, ce qui diffère un peu de la théorie de Mc Dermott où les relations primitives sont $<$,$>$ et $=$. Mais le pouvoir d'expression est en fait le même puisque avec au maximum deux arcs orientés entre deux nœuds on peut exprimer toutes les relations possibles les liant.

Une boucle constituée uniquement d'arcs ≤ peut en fait être réduite à un seul instant donc à un seul nœud. En revanche, si elle contient un arc ≠ alors elle est incohérente. Une fois ces réductions faites, les relations ≠ peuvent être implicites puisqu'elles ne sont pas propagées et sont donc supprimées : le réseau ne contient que des relations ≤ et n'a aucune boucle. On obtient ainsi un treillis temporel où la relation entre deux éléments est donnée par un chemin les reliant.

Pour faciliter la recherche, on transforme ce treillis en un arbre indexé. On choisit une racine dans le treillis et on ordonne les éléments suivant leur rang, défini comme la taille du plus long chemin les reliant à la racine. Ensuite, on place dans l'arbre un élément de rang maximal M et on lui choisit comme parent un élément parmi ses prédécesseurs de rang M-1, et ainsi de suite jusqu'à la racine. On recommence l'opération en repartant d'un nouveau nœud de rang maximal. S'il y a plusieurs choix possibles pour le parent d'un nœud, on en prend de préférence un qui ne figure pas encore dans l'arbre et s'il n'y en a pas on prend celui qui a le moins de fils dans l'arbre. Cet arbre a un nombre maximum de branches et est indexé. Sur la première branche, l'index correspond au rang. Soit $I(u) = [i_1.i_2...i_k]$ l'index du nœud u et $|s(u)|$ le nombre de successeurs de u, et $v \in s(u)$.

Si $|s(u)| = 1$ alors $I(v) = [i_1.i_2...i_{k-1}.(i_k+1)]$

Si $|s(u)| = 2$ alors $I(v) = [i_1.i_2...i_k.1]$

Si $|s(u)| > 2$ alors $I(v) = [i_1.i_2...i_k.(3-|s(u)|).1]$

Voici sur un exemple le passage d'un treillis à un arbre indexé :

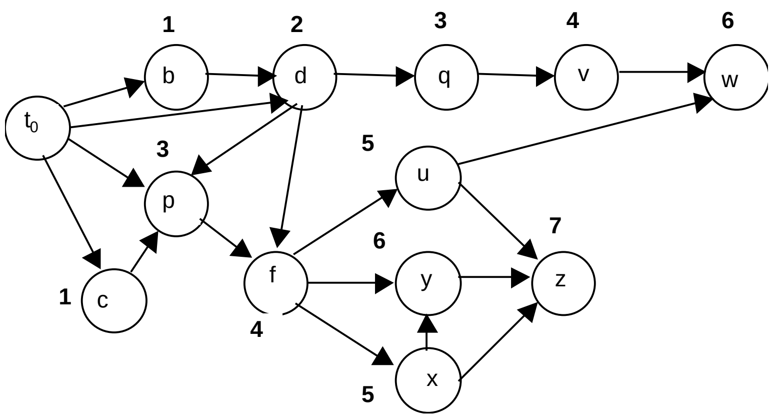


Fig. 17 : exemple : treillis initial avec les rangs

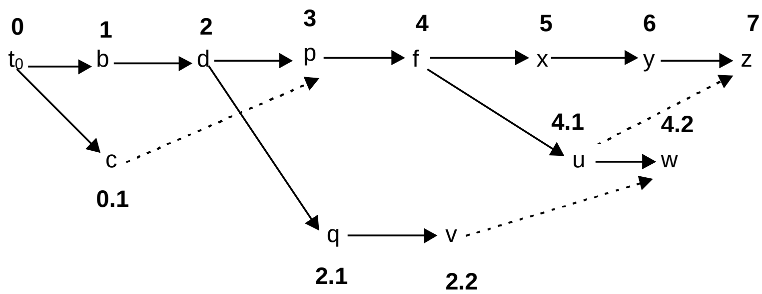


Fig. 18 : arbre final avec index

Les opérations habituelles d’un TMM sont effectuées ici en comparant les index ou en modifiant et réindexant l’arbre. On pourrait étendre ce TMM à des relations numériques entre dates ponctuelles et introduire la notion d’intervalle en étiquetant les arcs par des durées.

II.2.4 Systèmes d'inéquations

Lorsque l’on ne choisit pas de représenter les notions temporelles par des ensembles flous, la flexibilité n’est possible que si l’on utilise des fenêtres temporelles. C’est le choix fait par Fall [Fall 86] et Vere [Haton 91] ainsi que dans la version initiale de la technique des gabarits. Chaque événement possède une durée minimum et une durée maximum, ce qui permet, par propagation de contraintes, de déterminer une fenêtre temporelle délimitée par des dates de début au plus tôt, au plus tard, fin au plus tôt et au plus tard. Aucune date n’a la préférence à l’intérieur de cette fenêtre.

Le raisonnement consiste à restreindre ces fenêtres temporelles en vérifiant un système d'inéquations. Une activité est caractérisée par les paramètres suivants :

– début au plus tôt et au plus tard : d_{min}, d_{max}

– durée minimale et maximale : du_{min}, du_{max}

– fin au plus tôt et au plus tard : f_{min}, f_{max}

Les inéquations à vérifier sont :

- $f_{min} \geq d_{min}+du_{min}$
- $f_{max} \leq d_{max}+du_{max}$
- $d_{min} \geq f_{min}-du_{max}$
- $d_{max} \leq f_{max}-du_{min}$
- $du_{min} \geq f_{min}-d_{max}$
- $du_{max} \leq f_{max}-d_{min}$

On obtient une série d'inéquations similaires pour les contraintes de délai. Pour un délai entre un événement 1 et un événement 2, il suffit de remplacer les durées par les délais, les débuts par les débuts ou fins de l'événement 1 (suivant que le délai est relatif au début ou à la fin de l'événement) et les fins par les débuts de l'événement 2.

La propagation des contraintes temporelles consiste à appliquer ces inéquations tant que des valeurs sont modifiées. Ainsi on restreint les fenêtres temporelles en jouant sur les dates de début au plus tôt, début au plus tard, fin au plus tôt et fin au plus tard. L'algorithme s'arrête lorsque toutes les inéquations sont vérifiées.

II.2.5 Approche géométrique

J. F. Rit a élaboré une représentation géométrique de ce type de problèmes[Haton 91]. Il travaille dans l'algèbre d'intervalles avec des dates de début et de fin d'événements données par des fenêtres temporelles et des durées imprécises.

Un intervalle est vu comme un couple d'instants de début et de fin, la date de début devant se situer avant le date de fin. On peut représenter un intervalle et toutes les relations d'Allen qui le concernent dans le plan. Un point du plan donne une date de début et une date de fin. La date de début doit être avant la date de fin, ce qui correspond au demi-plan limité par la droite y=x.

Soit un point A du plan tel que sur le schéma suivant. Pour n'importe quel autre point B du plan correspondant à un intervalle, suivant la position de B par rapport à A on peut donner la relation primitive d'Allen qui lie A et B. Sur notre schéma par exemple, A est "pendant" (d) B puisque B débute avant A et se termine après. Par rapport à A, le demi-plan est en fait délimité en secteurs qui correspondent aux primitives d'Allen. Comme on le voit sur le schéma, la demi-droite horizontale et à gauche de A correspond à A "termine" (f) B.

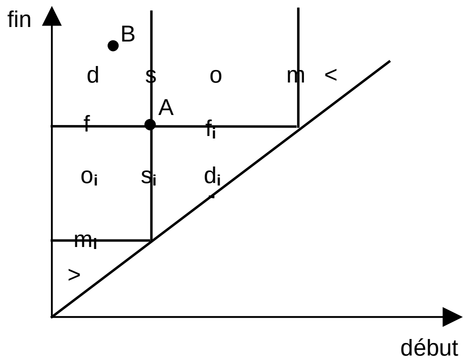


Fig. 19 : représentation planaire des relations d'Allen

On peut alors voir la conjonction de contraintes temporelles comme l'intersection de régions permises par chaque contrainte, et la disjonction de contraintes comme l'union de ces régions. Tout le raisonnement correspond à des opérations géométriques. Il y a incohérence lorsque pour un intervalle donné la propagation des contraintes conduit à un domaine du plan vide.

Voici une représentation géométrique des éléments temporels les plus courants :

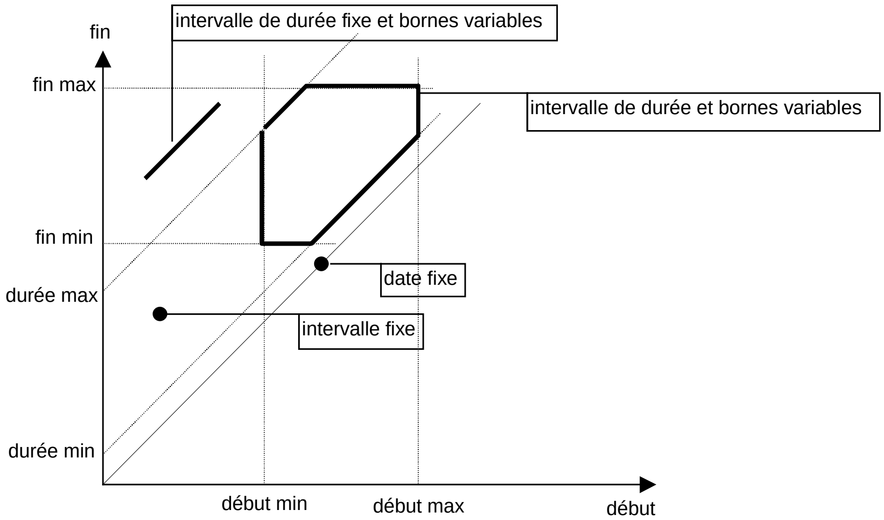


Fig. 20 : les différents objets temporels dans le plan

Cette approche est équivalente à la théorie d'Allen. Les raisonnements effectué sont identiques, cependant dans certains cas elle peut donner une représentation plus synthétique des éléments temporels. L'intérêt de cette représentation ne se situe pas dans la phase de raisonnement, qui est celle qui nous intéresse, mais plutôt dans la phase ultérieure de présentation des résultats.

II.2.6 Conclusion

Du point de vue des modélisations, les représentations mixtes offrent un pouvoir d'expression très large en autorisant aussi bien des contraintes numériques et symboliques. Le passage à des contraintes floues ne perd rien en pouvoir d'expression et permet de travailler sur un formalisme unifié : il n'y a plus de distinction numérique/symbolique. De plus, on peut ainsi introduire de la souplesse au niveau de l'évaluation de la satisfaction des contraintes.

Du point de vue des raisonnements, les graphes semblent être un outil très puissant et permettent de répondre à la plupart des questions que l'on se pose face à un problème temporel (consistance, relation entre deux événements…). Les techniques associées aux CSP sont très générales, elles englobent tous les types de modélisations. On peut même les étendre aux modélisations floues comme on le verra en III.1 p.73.

Avec les modélisations floues, les types de raisonnements rencontrés en dehors des différentes extensions des CSP reposent sur des formalismes logiques et la théorie des possibilités [Dubois 91], [Steunou 92]. Ces formalismes sont en dehors de notre périmètre de recherche puisqu'ils ne permettent pas de raisonner sur les données temporelles indépendamment des événements qui leur sont attachés. Ils s'intéressent à des relations de causalité où chaque événement est associé à un intervalle de validité temporelle. Les relations de causalité sont elles aussi associées à intervalle de validité. Les raisonnements sont du type : "A cause B est vraie sur l'intervalle de temps I et A est vraie sur I' donc B est vraie sur I" ". On considère en fait des intervalles de validité temporelle mais pas des relations temporelles entre événements.

Dubois, Steunou et Chen [Dubois 91], [Steunou 92] et [Chen 93] proposent tous trois l'évaluation d'expression complexes comportant des informations temporelles. Chen évalue par exemple les degrés de possibilité et de nécessité d'une expression comme "A dure longtemps et B a lieu après A", connaissant les distributions de possbilité des dates d'occurrence de A et B". Dubois et Steunou s'intéressent plutôt aux intervalles de validité d'expressions complexes. Connaissant les distributions de possibilité de deux faits A et B en fonction du temps ils en déduisent les intervalles sur lesquels la proposition "A et B" est vraie. Dans ces deux cas il n'y a pas de propagation des informations temporelles.

Chen réalise aussi un raisonnement par propagation de contraintes temporelles, mais les relations qu'il propage ne sont pas floues, ce sont les relations symboliques classiques auquelles il attache en plus des degrés de possibilité et de nécessité. Par rapport à la théorie d'Allen ou à celle de Mc Dermott il ajoute simplement une propagation des degrés de possibilité et nécessité qui permettent de prendre en compte l'incertitude des relations. Ceci ne répond pas à notre problème puisque nous cherchons à propager de l'imprécision et pas de l'incertitude.

Les seules méthodes qui permettent de propager des relations temporelles floues sont donc fondées sur les techniques des CSP. Nous les détaillons dans la présentation de l'extension de ces techniques à des contraintes floues en III.1.4A) p. 79.

II.3 Techniques de reconnaissance

Nous allons maintenant voir quelles sont les différentes techniques utilisées pour la reconnaissance de scenarii. Ces techniques sont utilisées dans plusieurs domaines, avec principalement la reconnaissance du dialogue, le diagnostic en général et la supervision [PRC-GDR 95]. La reconnaissance de scenarii repose sur un modèle de supervision associatif, par opposition aux modèles prédictifs qui simulent le comportement du système et aux modèles explicatifs qui établissent des liens entre les effets observés sur le système et leurs causes [Basseville 96]. Les scenarii décrivent quelques trajectoires du système dynamique pertinentes pour l'utilisateur. Le système de reconnaissance doit, à partir de l'observation de ce même système en fonctionnement, reconnaître le ou les scenarii qui se déroulent.

La reconnaissance peut déclencher des actions, des alarmes ou une représentation claire de la situation pour fournir éventuellement une aide à la décision. Dans le cadre du diagnostic de pannes, un fonctionnement normal est modélisé par des scenarii normatifs et les pannes par des scenarii de dysfonctionnement. La plupart du temps, la reconnaissance d'un scénario de dysfonctionnement déclenche une alarme.

Un scénario est une succession d'événements et de valeurs associées à des paramètres décrivant le système. Parmi tous ces paramètres, le temps joue souvent un rôle primordial, comme on l'a vu pour les gabarits. Etant données les propriétés discriminantes du temps on lui accorde un statut particulier. On parle alors de "scenarii temporels". La prise en compte du temps permet aussi d'effectuer des prédictions datées par rapport au processus observé.

Avant de commencer la reconnaissance, il faut constituer une bibliothèque de scenarii qui regroupe l'ensemble des situations caractéristiques que l'on souhaite effectivement reconnaître. La représentation des scenarii dépend du domaine modélisé et du mécanisme de reconnaissance choisi. Ici nous nous intéressons uniquement à un mécanisme de reconnaissance incrémental, c'est-à-dire que l'on cherche à reconnaître le scénario avant même qu'il n'ait fini de se dérouler, au fur et à mesure des observations que l'on peut effectuer.

Chaque élément d'observation est comparé aux éléments de scenarii et on essaye ainsi de les apparier. Pour unifier une observation avec un élément de scénario, il faut d'une part que l'observation et l'élément soient du même type (représentent un même événement), et d'autre part que l'observation vérifie toutes les contraintes qui, dans le scénario, la lient avec d'autres faits. Ces contraintes sont principalement temporelles dans le cas qui nous intéresse.

Nous allons présenter les différentes stratégies retenues pour la reconnaissance de scenarii temporels. Certaines sont fondées sur la vérification des contraintes temporelles alors que pour d'autres cette vérification n'est que secondaire et le mécanisme de reconnaissance s'appuie essentiellement sur des inférences logiques.

II.3.1 Reconnaissance par inférence logique

A) Formalisme de Kautz

a) Représentation des scenarii

Le formalisme de Kautz [Kautz 86] repose sur une décomposition hiérarchique des actions et une inférence logique du type déductif. Il a déjà été implémenté dans le cadre de la reconnaissance de dialogues.

La représentation des scenarii dans le formalisme de Kautz consiste en une décomposition hiérarchique des plans en forme d'arbre. La racine est l'objectif ou le but du plan et les feuilles sont des événements observables directement. La décomposition hiérarchique repose sur deux relations principales : la spécialisation et la décomposition. Il y a spécialisation entre une action particulière et une autre plus générale qui la subsume. L'action *faire des pâtes* est par exemple une spécialisation de *préparer un plat*. Les relations de spécialisation sont marquées en pointillés sur le schéma ci-dessous.

Une action est liée par une relation de décomposition à toutes les sous-actions qui sont nécessaires pour sa réalisation. *Faire des pâtes* se décompose en *faire bouillir de l'eau, faire des nouilles* et *faire de la sauce*. La décomposition est représentée par un trait plein. Dans [Hong 93], l'auteur fait apparaître une autre forme de relation, la relation de permission. L'action de *mettre la clé dans le démarreur* rend possible le fait de *démarrer,* il y a là une relation de permission.

Voici un exemple de représentation d'un scénario dans ce formalisme :

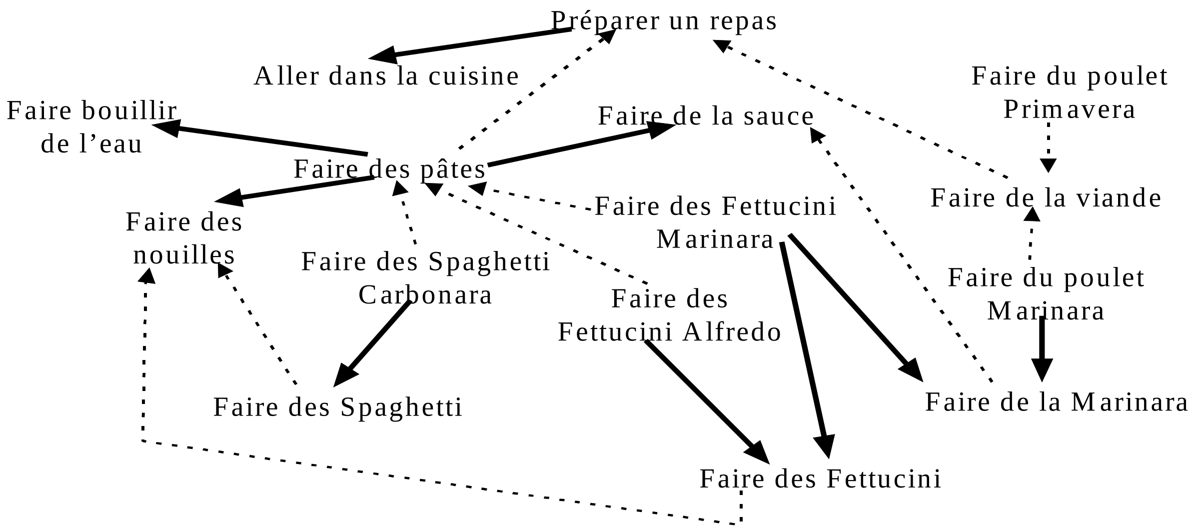


Fig. 21 : exemple de scénario dans le formalisme de Kautz

Cette représentation n'est pas encore suffisante. Il faut aussi décrire les relations temporelles qui existent entre les éléments. On utilise pour cela les relations d'Allen [Allen 81] détaillées en II.1.1A) p.17. Les observations sont des actions typées et datées, ce qui permet de les associer aux éléments de scenarii qui leur correspondent et de mettre à jour les contraintes temporelles.

L'axiome d'exclusion fait lui aussi partie de la modélisation des scenarii mais il n'apparaît pas sur la décomposition hiérarchique. Il stipule qu'une même action ne peut pas être en même temps une instance de deux actions différentes qui sont composées de cette action élémentaire. *Faire des Fettucini* ne peut pas être à la fois un élément de *faire des Fettucini Alfredo* et *faire des Fettucini Marinara*.

b) Reconnaissance structurelle

Le principe de la reconnaissance repose sur une ontologie de l'action, c'est-à-dire une description exhaustive des actions sous la forme de règles logiques. Ces règles se déduisent de la décomposition hiérarchique (chaque relation se traduisant par un axiome), des relations temporelles, de l'axiome d'exclusion et de l'hypothèse du monde clos. Cette dernière hypothèse a deux conséquences : un plan a une unique décomposition sous forme hiérarchique, il n'y a qu'une façon de le réaliser, et toutes les actions ont un but, elles doivent obligatoirement participer à un des plans modélisés. Tous les scenarii sont donc précompilés en tenant compte de ces hypothèses, et on obtient ainsi une base de règles.

A partir de cette base, la reconnaissance consiste à effectuer des inférences logiques déductives tout en intégrant les observations d'événements réels. C'est un raisonnement spécialisé qui fonctionne en "chaînage avant" avec une heuristique. L'heuristique consiste à garder un minimum d'hypothèses de plans, donc à regrouper le maximum d'observations sous un même plan. Pour chaque observation, on cherche les différents buts possibles en remontant les plans jusqu'à leur racine à l'aide des relations de décomposition et de spécialisation. Pour plusieurs observations, on choisit un ensemble minimal de buts possibles et on redescend dans la hiérarchie pour dériver les assertions qui en découlent. S'il y a inconsistance, on refait l'opération après un retour arrière en partant d'un plus grand ensemble de buts possibles. Ces opérations forment une opération de fermeture sur la hiérarchie d'actions.

Hong [Hong 93] raisonne lui de façon un peu différente à partir d'une représentation semblable bien qu'elle comporte une relation de plus, la relation de permission. Il aborde le problème de la reconnaissance plus comme la résolution d'un problème de satisfaction de contraintes. Ces contraintes sont à la fois structurelles et temporelles. Les axiomes obtenus de la hiérarchie constituent le mécanisme de propagation des contraintes. On obtient après propagation un ensemble de plans possibles avec les contraintes structurelles mises à jour et un marqueur indiquant pour chaque action si elle a été observée, si elle est justifiée par une autre action ou si elle est attendue. La reconnaissance se termine lorsque tous les éléments ont été justifiés. Cette méthode permet d'avoir une vision plus claire de l'état de la reconnaissance à chaque instant et par conséquent permet de fournir plus facilement des explications.

c) Reconnaissance temporelle

Initialement, la reconnaissance temporelle selon le formalisme de Kautz était fondée sur un algorithme de type CSP de propagation des contraintes. A chaque instanciation d'un fait, on ajoute une contrainte temporelle correspondant à sa période d'occurrence et on propage cette contrainte en utilisant les règles de composition d'Allen jusqu'à ce qu'il n'y ait plus de modification. Ceci permet d'éliminer certains plans candidats qui ne vérifieraient pas les contraintes temporelles.

Kautz rend ensuite la datation des événements observés un peu plus flexible [Kautz 91.2]. Au lieu de faire correspondre à une observation un intervalle d'occurrence précis, il lui associe un intervalle de temps "flou" ("flou" n'est pas entendu ici au sens de la théorie des sous-ensembles flous). Il n'est en effet pas toujours possible de dater précisément un événement. Un intervalle imprécis <a,b,c,d> indique que l'événement débute entre a et b et se termine entre c et d. On se situe toujours dans le cadre de la théorie d'Allen, et il faut ajouter un certain nombre d'axiomes pour propager de telles informations temporelles. Un de ces axiomes est :

Si T1 = <a,b,c,d> et T2 = <e,f,g,h> et T1 {si} T2 (T1 débuté par T2)

Alors T1 = <max(a,e),min(b,f),max(c,g),d>

Song et Cohen proposent un renforcement de l'algorithme de propagation spécifique à l'action de reconnaissance suivant le formalisme de Kautz [Song 91]. Ils font pour cela une hypothèse supplémentaire qui coïncide avec l'hypothèse du monde clos. Cette hypothèse dit que toute action est totalement déterminée et bornée dans le temps par ses sous-actions, elle doit donc débuter avec sa première sous-action et se terminer avec la plus tardive. Si on fait cette hypothèse supplémentaire, il apparaît que l'algorithme d'Allen n'est plus assez restrictif comme le montre l'exemple suivant.

A est décomposée en deux sous actions a_1 et a_2 avec a_1 avant a_2 et a_1 (de même pour a_2) {débute (s) ou finit (f) ou pendant (d) ou égale (eq)} A. Si on tient compte en plus de l'hypothèse, une des deux sous-actions au moins doit débuter en même temps que A et une des deux sous-actions au moins doit se terminer en même temps que A. On obtient alors des contraintes plus précises, a_1 étant la seule sous-activité qui puisse débuter en même temps que A et a_2 la seule qui puisse se terminer en même temps que A.

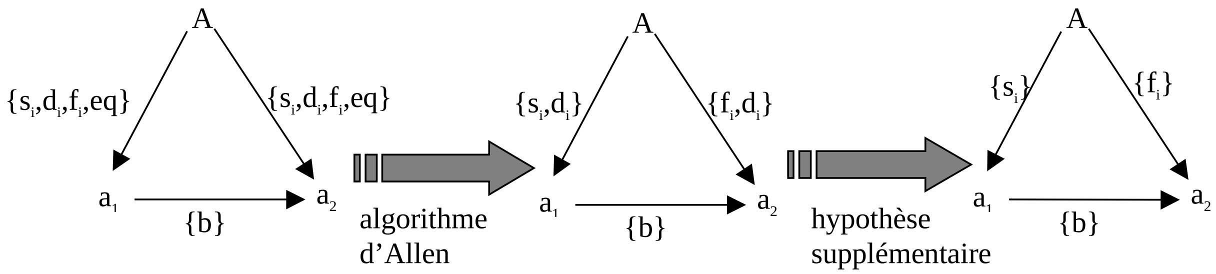


Fig. 22 : effet de l'hypothèse supplémentaire

On peut décrire un certain nombre de cas semblables où l'algorithme d'Allen se révèle insuffisant. On dénombre 5 sous-cas classés suivant la contrainte liant a_1 à a_2 dans le cas d'une décomposition en deux sous-actions. Pour plus de deux sous-actions, on se ramène aux sous-cas précédents en introduisant des sous-actions fictives. L'algorithme renforcé de reconnaissance se présente ainsi :

- 0. Algorithme d'Allen (PC1 sur algèbre d'intervalles)
- 1. Algorithme de Kautz puis mise à jour des contraintes temporelles avec le renforcement de l'algorithme temporel
- 2. Retour à 0 et 1 tant que les contraintes temporelles sont modifiées

L'algorithme est ainsi plus restrictif et permet une reconnaissance plus discriminante donc moins coûteuse en gestion d'hypothèses.

B) Reconnaissance abductive

a) Contexte opérationnel

Nous allons présenter ici deux systèmes de reconnaissance qui fonctionnent sur des principes très similaires et dans des domaines voisins. Ils opèrent tous deux dans le domaine de la reconnaissance militaire et sont intégrés dans des systèmes dits de "**C**ommande, **C**ontrôle, **C**ommunication et **I**ntelligence" (C^3I).

Le système MATIS (**M**odèle d'**A**ide à la décision par le **T**raitement de l'**I**nformation **S**ymbolique) travaille à partir d'observations fournies sur le terrain pour aider à la reconnaissance d'une situation tactique et expliquer l'attitude ennemie en détectant les buts poursuivis [Barès 94], [Cañamero 94]. Le système SCAN (**SC**ript-**AN**alyst) a les mêmes fonctions mais est spécialisé dans le domaine terre [Lakowski 87].

On fait deux hypothèses simplificatrices pour la reconnaissance. L'ennemi est censé poursuivre un objectif et donc ses actions sont toutes explicables. Il est aussi considéré comme un agent rationnel, c'est-à-dire qu'il effectue des sous-actions afin de réaliser un plan global.

b) Représentations

Les deux systèmes utilisent une bibliothèque de scenarii qui regroupe les schémas classiques d'activités et de tactiques militaires. Mais les scenarii sont représentés de façon différente dans les deux cas, bien qu'ils contiennent finalement le même type d'informations. La forme de la représentation influe aussi sur le mécanisme de reconnaissance. Nous allons donc voir les deux formes de représentations de manière distincte.

MATIS utilise une description des activités sous forme de XPlans, qui sont dérivés des frames de Schank. Un XPlan est défini par : un schéma, des paramètres, des préconditions, un algorithme de planification et un algorithme de reconnaissance, des post-conditions et des motivations. Prenons comme exemple l'observation de l'activité d'un conducteur, on peut représenter l'action de doubler une voiture par un XPlan.

Le schéma est une description informelle de l'activité modélisée. Les paramètres sont tous ceux qui sont utiles à la reconnaissance, comme la vitesse de la voiture, sa distance à la voiture de devant. Les préconditions constituent les conditions nécessaires pour réaliser une activité. Dès qu'elles sont réunies, on déclenche le plan. Une des préconditions pour doubler est par exemple d'être sur une route à deux voies sans ligne blanche continue.

L'algorithme de planification décrit la séquence d'actions à réaliser pour atteindre le but du XPlan du point de vue d'un agent qui agirait "normalement": accélérer jusqu'à une distance raisonnable de la voiture que l'on veut doubler, se mettre sur la file de gauche... De là on doit déduire l'algorithme de reconnaissance qui donne en fonction des paramètres observés le degré de confiance à associer au scénario et les différents XPlans à activer (ex : si la voiture se rapproche de la voiture précédente, augmenter la confiance dans le XPlan "doubler une voiture").

Les post-conditions déterminent soit à quel moment on peut considérer qu'un plan a été réalisé (post-conditions de réussite), par exemple lorsque la voiture se trouve devant celle qu'elle doublait, soit à quel moment on doit l'abandonner (post-conditions d'échec). Elles déclenchent la désactivation du plan dans les deux cas. Enfin, les motivations du plan sont tous les éléments qui peuvent fournir une explication sur l'attitude observée, que ce soit les buts du conducteur (gagner du temps) ou le contexte (voiture devant)...

On distingue deux types de XPlans : ceux qui ne dépendent pas de la décision de l'agent, ils sont dits génériques (approcher d'un feu rouge), et ceux qui font intervenir un choix de l'agent. On regroupe ces XPlans dans une hiérarchie sous la forme d'un arbre disjonctif (cf. figure ci-dessous).

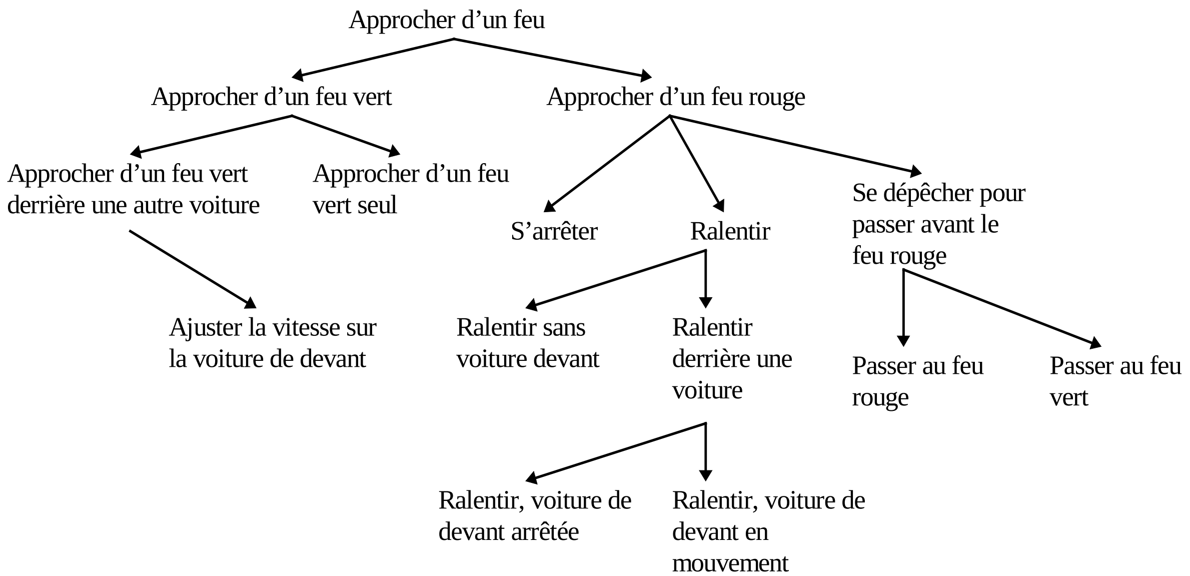


Fig. 23 : hiérarchie de Xplans

SCAN utilise lui des descriptions de scenarii sous forme de scripts. Cette description diffère assez peu des XPlans. Un script contient une liste d'éléments, c'est-à-dire la liste des autres scripts ou des propositions intervenant dans le script. Ces éléments seront vus comme des requêtes, et nous verrons le rôle des requêtes dans le mécanisme de reconnaissance.

Les liens d'un script permettent de décrire son contexte d'occurrence en donnant un certain nombre de variables et éventuellement leurs valeurs. Les préconditions sont décomposées en deux ensembles : les préconditions nécessaires ou statiques et les préconditions suffisantes ou dynamiques. Les préconditions statiques sont des propositions dont la valeur de vérité ne change pas dans le temps, contrairement aux préconditions dynamiques qui dépendent du contexte présent. Les préconditions sont des propositions et des contraintes temporelles entre éléments qui doivent être vérifiées pour instancier un script. On examine d'abord les conditions nécessaires, et si elles ne sont pas vérifiées il est inutile de regarder les conditions suffisantes. Enfin, l'analyse d'un script est une fonction d'évaluation qui permet de calculer la vraisemblance du script en fonction des paramètres temporels et autres.

c) Reconnaissance

Dans le cadre des XPlans, la reconnaissance s'appuie sur une base de règles et d'algorithmes traduits des XPlans. La reconnaissance est un parcours en profondeur d'abord de la hiérarchie des XPlans. Lorsque l'on atteint une feuille de l'arbre, cela signifie que l'on a reconnu une séquence complète.

L'algorithme se décompose ainsi :

- 0. La racine de l'arbre est active.
- 1. Examiner les post-conditions des XPlans actifs dans l'ordre décroissant des degrés de confiance.
- 2. Si les post-conditions de réussite sont satisfaites, le plan est réussi et désactivé.
- 3. Si les post-conditions d'échec sont satisfaites, le plan échoue et est désactivé.
- 4. Sinon, on examine les préconditions des XPlans fils :
- 5. Si elles sont vérifiées, activer le plan et désactiver le XPlan père.
- 6. Sinon rien.
- 7. Pour le nouvel ensemble de XPlans actifs :
- 8. Appliquer l'algorithme de reconnaissance de chaque plan et activer les XPlans spécifiés dans cet algorithme.
- 9. Si la reconnaissance échoue, diminuer la confiance dans le plan d'une quantité prédéfinie.
- 10. Si tous les XPlans d'une branche ont été désactivés, réactiver la racine. Retour en 0.

Cet algorithme donne une vision globale de la stratégie suivie mais n'exprime pas exactement l'ordre des opérations tel qu'il est. On favorise en fait le développement en profondeur d'abord de la branche la plus prometteuse, c'est-à-dire celle qui a le plus grand degré de satisfaction. Il n'y a pas de traitement particulier des contraintes temporelles, celles-ci peuvent apparaître comme de simples paramètres avec des prédicats liant ces paramètres.

La reconnaissance à partir de scripts fonctionne comme un système expert en chaînage arrière guidé par les requêtes. On examine pour chaque script ses préconditions. Dès que ces préconditions sont vérifiées, le script est instancié et on essaye de reconnaître les différents éléments de sa liste : c'est en ce sens que la recherche est guidée par les requêtes. On calcule ensuite la vraisemblance du script par son analyse qui dépend de la vraisemblance des éléments de sa liste, de la vérification des relations temporelles, de l'histoire des valeurs de vraisemblance et de l'élément en cours d'instanciation. Les poids de ces différents facteurs sont spécifiés dans la fonction d'analyse.

Les contraintes temporelles apparaissent de façon explicite dans les préconditions et de façon implicite dans la liste d'éléments : les éléments sont ordonnés suivant leurs dates de début dans la liste. Les éléments temporels sont ceux issus de la théorie d'Allen auxquels on ajoute la notion de durée d'un événement et les relations entre dates de début des événements. Les contraintes temporelles interviennent uniquement au niveau du calcul du degré de vraisemblance du script. Pour ne pas considérer trop d'hypothèses simultanément, on élimine toutes celles dont le degré de vraisemblance est inférieur à un certain seuil.

Dans les deux cas on peut parler de raisonnement abductif dans la mesure où l'on fait des hypothèses sur les causes à partir des observations qui sont des effets supposés de ces causes. Par exemple, si l'on observe une accélération d'une voiture, on émet l'hypothèse d'un dépassement alors qu'en logique déductive c'est le dépassement qui implique une accélération. On ne formule donc que des hypothèses qui sont confirmées ou infirmées par la suite. Il faut pouvoir comparer les différentes hypothèses et donc pour cela les évaluer en leur attribuant des degrés de vraisemblance que l'on ne trouve pas dans le formalisme de Kautz.

C) Automates à états finis

a) Classification des scenarii

L'approche de Lévy est fondée sur l'algorithme KMP de Knuth, Morris, et Pratt pour la reconnaissance de motifs dans une chaîne de caractères [Lévy 95]. Cette méthode est présentée dans un cadre général et il n'existe pas, à notre connaissance, d'application réelle l'utilisant. Les scenarii sont représentés sous la forme d'automates à états finis. Les états sont les événements observés et les arcs sont étiquetés par les relations temporelles.

On peut modéliser différents types de scenarii par cette méthode mais il faut noter une restriction importante : les scenarii doivent impérativement être totalement ordonnés. Les scenarii sont donc des séquences d'événements non déterminés dans le temps mais dont l'ordre est lui fixé d'avance. On peut en fait traiter les scenarii partiellement ordonnés comme un ensemble de scenarii totalement ordonnés, mais cette méthode est très lourde et inefficace pour l'instant. On se limite donc à des scenarii totalement ordonnés.

Les scenarii peuvent aussi être linéaires ou non. Un scénario est linéaire si les contraintes temporelles n'apparaissent qu'entre des éléments successifs du scénario. Dans le cas de scenarii non linéaires, il faudra enregistrer les informations relatives au passé pour pouvoir vérifier les contraintes temporelles.

On parle aussi de scenarii exclusifs et de scenarii extraits. Pour un scénario exclusif, aucun élément ne peut s'intercaler entre ceux prévus par l'automate. Il n'y a en fait aucun degré de liberté et ceci implique une description précise et statique des scenarii. Si on souhaite un peu plus de souplesse dans la reconnaissance on peut utiliser des scenarii extraits. On ne tient alors pas compte de toutes les observations pour la reconnaissance, mais seulement de celles qui correspondent à un état possible de l'automate.

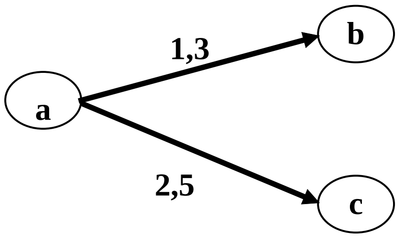


Fig. 24 : exemple d'automate partiellement ordonné

L'automate de la figure ci-dessus signifie que a est suivi de b avec un intervalle de temps de une à trois unités et d'une occurrence de c avec un intervalle de deux à cinq unités. a(i) signifie que l'événement a a lieu à la date i. {a(0),b(1),c(2)} est un scénario compatible avec l'automate ainsi que {a(0),c(2),b(3)}. L'automate est partiellement ordonné puisque les événements b et c ne sont pas ordonnés. Si l'automate est exclusif, il rejette le scénario {a(0),b(1),d(3),c(5)}alors que le scénario extrait du précédent {a(0),b(1),c(5)} est compatible avec l'automate.

Les scenarii les plus simples à traiter sont totalement ordonnés, linéaires et exclusifs mais pour plus de réalisme il faut aussi considérer d'autres formes de scenarii.

b) Scenarii linéaires, exclusifs et totalement ordonnés

Pour ce type de scenarii et si on supprime les contraintes temporelles on se ramène à la reconnaissance de mots par l'algorithme KMP. On prévoit de façon exhaustive tous les événements possibles. Voici un exemple d'automate pour reconnaître le mot "ananas" :

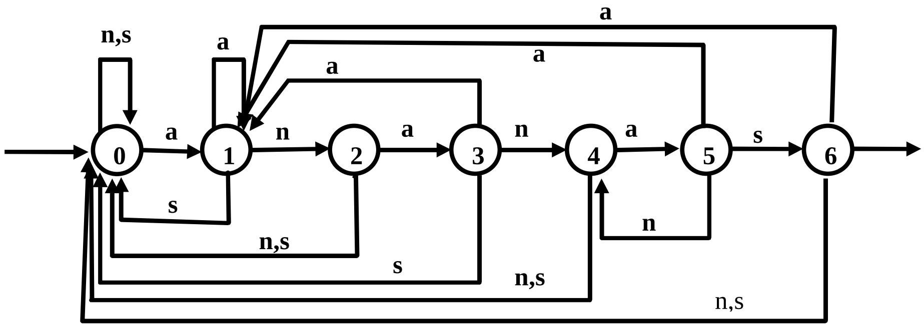


Fig. 25 : reconnaissance du mot "ananas"

Le mot est reconnu lorsque l'on arrive à l'état 6. Chaque lettre reconnue permet d'établir une transition d'un état à un autre. Si la lettre reconnue n'est pas celle attendue pour obtenir le mot choisi, on fait une transition vers un état précédent. Si on a déjà reconnu "anana" et que l'on reconnaît la lettre n, on ne reconnaît pas "ananas" mais "ananan" donc on ne passe pas à l'état 6. En supprimant les deux premières lettres du mot reconnu on obtient "anan" qui correspond à l'état 4 donc on a un retour arrière vers l'état 4. Ceci explique la présence d'une transition étiquetée "n" entre 5 et 4.

Si on veut modéliser des scenarii temporels avec des contraintes temporelles entre les états il faut ajouter dans les transitions les contraintes temporelles. a(2,5) signifiera par exemple que l'événement a est attendu mais uniquement sur un intervalle de temps de 2 à 5 unités après l’instant où l’on a atteint l’état précédent. Si a arrive en dehors de ces limites, on aura une transition a' qui oblige à un retour arrière.

Mais cette opération va décaler tous les états d'un certain nombre de crans, et il faut vérifier de nouveau toutes les contraintes temporelles. On devra donc garder en mémoire les dates d'occurrence de tous les événements pour pouvoir faire cette vérification en cas de retour arrière. Sur la figure suivante cette date apparaît dans un rectangle dans le cercle de l'état correspondant. On fait l’hypothèse que les seuls événements observables sont a, b ou c.

Voici l'automate que l'on obtiendrait pour des observations : {a(0),b(3),a(5),b(6)}.

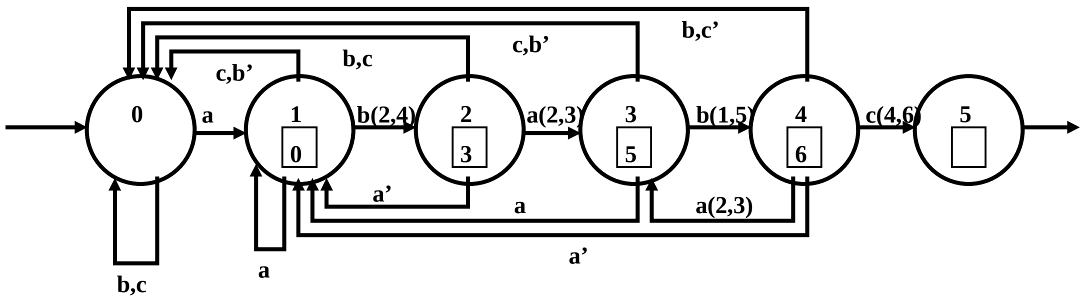


Fig. 26 : reconnaissance datée du mot "ananas"

On a atteint l'état 4 de l'automate. Si on observe ensuite c(11), alors on a bien la transition de l'état 4 à 5 puisque c arrive 5 unités de temps après b, ce qui est bien entre 4 et 6. Le scénario est alors reconnu. Supposons qu'au lieu de c(11) on ait c(17), alors les contraintes temporelles ne sont plus respectées et on a une transition c' avec un retour à l'état 0.

Revenons à l'état 4. Si l'on observe a(10), on a une transition a' et on revient à l'état 1 sans avoir à vérifier aucune contrainte temporelle. Si au lieu de a(10) on observe a(9), alors la contrainte a(2,3) est bien vérifiée et on a un retour arrière vers l'état 3. Il faut décaler de 2 crans toutes les dates associées aux états. La date 6 anciennement affectée à l’état 4 se retrouve associée à l’état 2 et ainsi de suite, ce qui nous donne :

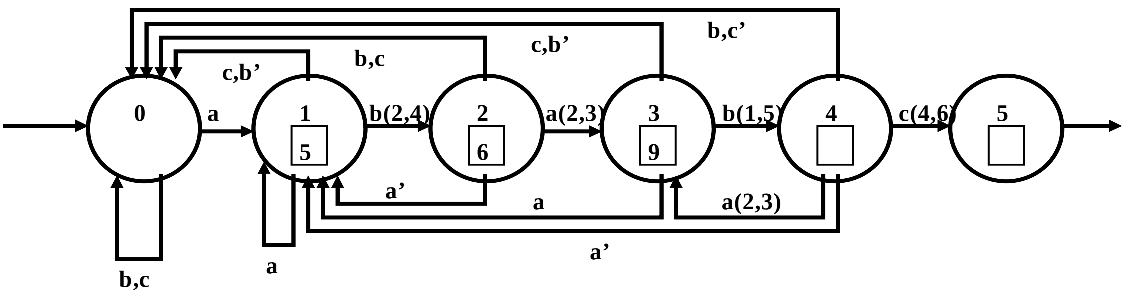


Fig. 27 : retour arrière avec contraintes temporelles

Mais alors la contrainte temporelle entre l'état 1 et 2 n'est pas vérifiée et on revient à l'état 1 en effectuant une transition a' depuis l'état 4 comme si l'occurrence de a ne respectait pas la contrainte temporelle a(2,3). La séquence est réduite à {a(9)}.

Sur un scénario de longueur importante, il peut être très coûteux de vérifier de nouveau toutes les contraintes temporelles à chaque retour arrière, surtout lorsque les échecs de la reconnaissance sont plus souvent liés à la reconnaissance des séquences d’événements qu’à la satisfaction des contraintes. Dans ce cas là on peut se permettre de ne pas vérifier à nouveau les contraintes et de donner la priorité à la reconnaissance de la séquence.

Dans ce cas, une fois la reconnaissance terminée, il faut vérifier toutes les contraintes temporelles depuis l’état 0 jusqu’à l’état résultant du dernier retour arrière. Si elles ne sont pas satisfaites, on recommence la reconnaissance en partant du suffixe le plus long de la séquence pouvant correspondre avec l’automate. Une nouvelle fois on ne vérifie pas les contraintes tant que la reconnaissance n’est pas complète. Cette méthode dite "laxiste" n’est valable que si les cas d’échecs dus à la non satisfaction des contraintes temporelles sont rares.

c) Scenarii plus complexes

Dans la réalité, les scenarii sont plus souvent non linéaires, ils font intervenir des relations dites secondaires entre états non successifs. On peut ici aussi utiliser une méthode laxiste et ne vérifier les contraintes secondaires qu’en cas de reconnaissance du scénario complet. Sinon il faut garder en mémoire la contrainte secondaire au moment où on la rencontre pour la vérifier lorsque l’on atteint son état cible.

Soit l’événement a lié par la contrainte (2,3) à l’événement b. Lorsque a est reconnu et daté, a(5) par exemple, on peut en déduire que b doit arriver dans l’intervalle de dates [7,10]. C’est sous cette forme que l’on va garder l’information donnée par la contrainte. On associe à chaque état non encore reconnu un jeton contenant toutes ses contraintes temporelles sous la forme de tableau. La $i^{ème}$ ligne contient la contrainte temporelle avec le $i^{ème}$ état suivant, la première colonne donnant la limite inférieure et la deuxième la limite supérieure.

Lorsque l’on atteint un état, on modifie son jeton en remplaçant ses contraintes sur des durées par des contraintes de dates comme précédemment. Au fur et à mesure de la reconnaissance, on propage le jeton courant en supprimant sa première ligne (elle est vérifiée si on passe à l’état suivant) et en décalant toutes les autres d’un cran vers le haut.

Voici ce que cela donne sur un exemple traité étape par étape :

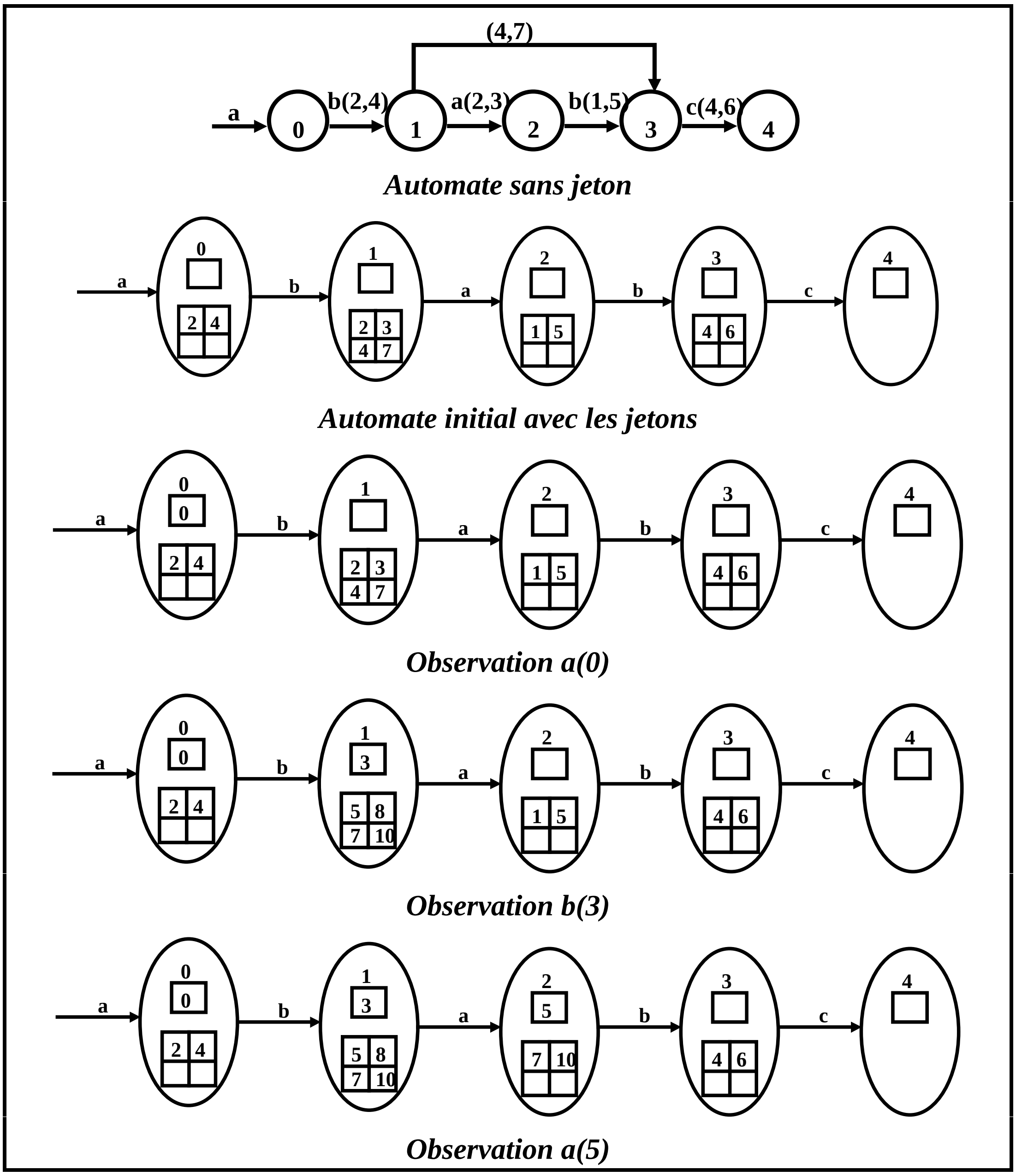


Fig. 28 : propagation de jetons dans un automate

Dans cette dernière étape on propage la ligne 2 du jeton de l'état 1 ([7,10]) sur la ligne 1 du jeton de l'état 2. A l'étape précédente, cette ligne contenait la contrainte sur les durées (1,5). Cette information se transforme en une contrainte sur les dates [6,10] sachant que a a lieu à la date 5. On intersecte les deux contraintes [7,10] et [6,10], ce qui nous donne [7,10]. On pourrait continuer la reconnaissance avec un événement b(8) par exemple.

Cette méthode devient complexe dès qu'il y a une incompatibilité. Il faut alors envisager de faire un retour en arrière, et il faut vérifier de nouveau toutes les contraintes et mettre à jour les jetons, ce qui nécessite de refaire tout le raisonnement sur les jetons depuis l'état 0.

Le problème devient aussi complexe et coûteux pour des scenarii non exclusifs. Une instance de scénario doit en effet être créée pour chaque séquence extraite correspondant à un préfixe du scénario. Chaque instance a ses propres jetons. La seule forme de rejet intervient lorsque l'on dépasse le délai et que l'élément attendu par l'automate n'est pas arrivé. Il n'y a jamais de retour arrière dans ce type de scénario, lors d'un rejet on efface purement et simplement l'instance du scénario. La reconnaissance reste faisable dans ces conditions tant que le nombre d'instances de scenarii candidates simultanément à la reconnaissance n'est pas trop élevé.

On peut aussi modéliser par des automates des scenarii comportant certains éléments facultatifs. C'est ce qui est fait dans [Azarewicz 86], mais il n'y a pas de contraintes temporelles exprimées dans les transitions. Un exemple de scénario de ce type est :

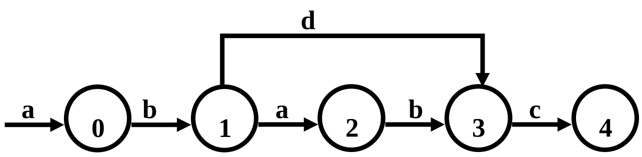


Fig. 29 : automate avec éléments facultatifs

Les transitions par les événements a et b peuvent être remplacées par une seule transition comportant l'événement d. L'état 2 est facultatif. On peut très bien imaginer d'utiliser aussi des contraintes temporelles, mais on retombe sur les mêmes problèmes en cas d'inconsistance et de retour arrière. Globalement, la technique des automates à états finis est simple et efficace lorsque les scenarii sont totalement ordonnés, linéaires et exclusifs mais devient lourde dès que les scenarii sont plus complexes.

D) Théorie du recouvrement parcimonieux

a) Reconnaissance abductive

La théorie du recouvrement parcimonieux (PCT : **P**arsimonious **C**overing **T**heory) est une technique utilisée pour le diagnostic médical [Wainer 98]. Elle procède par un raisonnement abductif. On établit en effet des relations de causalité entre maladies et symptômes. Ces relations de causalité sont des paires (maladie, symptôme). Les effets d'une maladie sont tous les symptômes liés à celle-ci par une telle relation.

D'autre part, on note l'ensemble des symptômes observés chez le patient. Un diagnostic ou une explication de ces symptômes est un ensemble E de maladies qui couvre tous les symptômes observés, c'est-à-dire que pour chaque symptôme s_i observé il existe au moins une maladie m_i dans E telle que l'on ait la paire (m_i,s_i). Cette définition seule conduit la plupart du temps à plusieurs diagnostics. On ajoute alors des propriétés supplémentaires. Une propriété classique est la minimalité de E : E est l'explication qui contient le minimum de maladies, tout sous-ensemble de E ne couvre pas la totalité des symptômes.

b) Modélisation temporelle floue

Une autre façon de discriminer les différents diagnostics possibles est d'ajouter des relations temporelles dans la description de l'enchaînement des symptômes, aussi bien pour les symptômes observés chez le patient que dans la description du déroulement d'une maladie. Ces descriptions sont souvent entachées d'imprécisions, dans ce cas on utilise des ensembles flous pour modéliser les relations temporelles [Wainer 98].

On ajoute dans la description d'une maladie les informations temporelles concernant les durées et les dates des symptômes ainsi que l'ordre d'enchaînement et les délais entre symptômes. On représente ceci par un arbre où les nœuds sont les symptômes associés à des durées, les arcs orientés donnent l'ordre d'apparition des symptômes et sont étiquetés par les délais. On représente de la même façon l'apparition des symptômes chez le patient avec en plus les dates floues d'apparition des symptômes.

On peut aussi ajouter une information dite "catégorielle" pour augmenter la discrimination entre les différents diagnostics possibles. A chaque symptôme on associe l'étiquette N s'il est nécessaire et P si son apparition est possible mais pas systématique dans l'évolution de la maladie. De même on décrit les symptômes présents et les symptômes dont on a vérifié qu'ils sont non présents chez le patient.

c) Raisonnement temporel

Le raisonnement temporel ainsi que le raisonnement sur la catégorie des symptômes interviennent dans une deuxième phase du raisonnement, une fois que l'on a sélectionné les explications qui couvrent les symptômes observés. Afin de classer ou même d'éliminer certaines explications, on utilise dans un deuxième temps l'information temporelle et catégorielle en calculant des indices de consistance.

Pour calculer la consistance temporelle il faut d'abord propager l'information dans l'arbre. Cette propagation se fait des feuilles de l'arbre vers sa racine et sans retour arrière. Les dates de début des symptômes correspondant aux nœuds feuilles sont fixées par l'observation. Pour les autres nœuds, on effectue une propagation. Pour les symptômes nécessaires mais réfutés jusqu'à la date du diagnostic on leur donne la date de début correspondant à "n'importe quand dans le futur". Pour tous les autres symptômes non observés on leur donne la date de début "n'importe quand".

Soit $BEG(s_i)$ la date de début observée pour le symptôme s_i, $BEG_{\leftarrow}(s_i)$ sa date de début après propagation et $DIST(s_i,s_j)$ le délai entre deux symptômes qui se suivent dans l'arbre. Alors pour les nœuds autres que les nœuds feuilles on a :

$$BEG_{\leftarrow}(s_i) = BEG(s_i) \cap \left[\bigcap_j \left\{ BEG_{\leftarrow}(s_j) - DIST(s_i, s_j) \right\} \right] \text{ avec les opérateurs flous.}$$

Un premier degré de consistance est alors donné par la hauteur de l'ensemble flou début du symptôme correspondant à la racine de l'arbre. Mais il faut aussi tenir compte des durées des symptômes. Pour chaque nœud on a un degré de consistance sur sa durée donné par la hauteur de l'ensemble intersection de la durée observée et de la durée attendue. Le degré de consistance global sur les durées est le plus petit des degrés individuels. Enfin le degré de consistance temporelle du diagnostic complet est encore le minimum des deux degrés de consistance sur les débuts et sur les durées.

Le temps intervient donc comme information supplémentaire pour éliminer certains diagnostics et pour classer les autres. Les informations temporelles permettent aussi de prévoir les dates des prochains symptômes normalement attendus en propageant les dates cette fois-ci de la racine vers les feuilles.

II.3.2 Reconnaissance fondée sur le temporel

A) Reconnaissance par comparaison de graphes

a) Représentation des graphes

Le principe de la méthode [Fontaine 93], [Fontaine 98] consiste à représenter le modèle de scénario que l'on cherche à reconnaître ainsi que les activités que l'on a réellement pu observer sous la forme de deux graphes : le graphe du scénario et le graphe de la session. Le résultat de la reconnaissance dépend de la comparaison de ces deux graphes, d'où le nom donné à cette méthode. Cette méthode a été appliquée au monitoring de patients en milieu médical [Dojat 98]. La modélisation retenue est une représentation par un graphe où les contraintes sont des intervalles numériques (cf. II.1.2A) p.19).

b) Propagation des contraintes temporelles

Avant de commencer la reconnaissance, on cherche le graphe minimal exprimant le scénario, c'est-à-dire le graphe n'excluant aucune solution mais ayant les contraintes les plus restreintes possibles. Le scénario est cohérent si le graphe minimal ne contient pas la contrainte vide. Pour obtenir ce graphe minimal, il faut propager les contraintes et les composer entre elles selon les techniques propres aux problèmes de satisfaction de contraintes. La définition exacte de minimalité ainsi que les algorithmes qui permettent de l'obtenir sont donnés en II.2.1 p.30.

c) Principe général de la reconnaissance

Comme nous l'avons dit en introduction, le principe de ce type de reconnaissance repose sur la comparaison de graphes, ce qui fait appel à un certain nombre de notions théoriques et d'algorithmes que nous donnons en annexe VII.1.1 p.239. Ici nous ne présentons que les idées générales du mécanisme. Il s'agit donc de comparer le graphe du scénario G(S) de contraintes r_{ij} et le graphe de la session $\Gamma(\Sigma)$ de contraintes ρ_{ij}.

On parle de reconnaissance globale par opposition à reconnaissance incrémentale lorsque la reconnaissance a lieu une fois que la session est terminée ; on intègre directement tous les éléments de la session et non pas un par un au fur et à mesure de leur arrivée. On peut avoir trois types de conclusions après une reconnaissance globale : la session satisfait le scénario, ce qui est une association forte, la session est compatible avec le scénario, ce qui est plus faible, ou enfin la session rejette le scénario si la reconnaissance échoue. On compare les deux graphes sous leur forme minimale et uniquement s'ils sont cohérents.

La notion de compatibilité résulte de la comparaison des solutions de chacun des graphes. Une session est compatible avec un scénario si au moins une de ses solutions est compatible avec le scénario, c'est-à-dire qu'elle peut être étendue en une solution du scénario. Suivant que les deux graphes n'ont aucun, quelques ou tous leurs nœuds communs on dit qu'ils sont compatibles séparément, partiellement ou totalement. La propriété de compatibilité n'implique pas que la séquence des événements dont la session constitue une trace vérifie effectivement le scénario.

On introduit aussi la notion de satisfaction, plus forte que la compatibilité. Il y a une certaine gradualité dans la notion de satisfaction et on donne 4 formes de satisfaction. La satisfaction simple est la notion la plus forte. La session Σ satisfait simplement le scénario S si et seulement si pour toute solution σ de $\Gamma(\Sigma)$, il existe au moins une solution s de G(S) compatible avec σ. On est alors certain que la séquence des événements tels qu'ils se sont effectivement déroulés pendant la session, et dont on n'a qu'une description incomplète ou imprécise, vérifie les contraintes imposées par le scénario. De même que pour la compatibilité, on peut nuancer la notion de satisfaction simple en ajoutant si elle est totale, partielle ou séparée. Les autres notions de satisfaction découlent de nuances sur la comparaison des graphes et sont moins utilisées.

Dans une deuxième phase on peut raffiner ces niveaux d'évaluation symboliques [Fontaine 98], par l'introduction de degrés de compatibilité ou d'incompatibilité numériques. Le degré de compatibilité est globale à la session et permet d'ordonner plusieurs sessions compatibles avec un même scénario. Ce degré résulte d'une interprétation probabiliste des intervalles de dates. En considérant les intervalles comme des distributions de possibilités uniformes, on calcule la probabilité qu'une solution de la session soit aussi une solution du scénario. Le degré d'incompatibilité est un peu similaire, il permet d'ordonner deux sessions toutes deux incompatibles avec le scénario. Le degré d'incompatibilité reflète une distance moyenne entre les solutions de la session et les solutions du scénario.

La reconnaissance incrémentale ne se fait plus à partir de la totalité de la session mais seulement avec une représentation partielle. On considère en fait des sessions Σ_k obtenues après introduction de k contraintes temporelles. La reconnaissance doit être dynamique, il ne s'agit pas seulement de juger de la situation relative d'un scénario et d'une session partielle, mais aussi d'estimer la possibilité de parvenir ultérieurement à une compatibilité ou à une satisfaction totale. On regarde en fait s'il existe un prolongement de la session partielle qui puisse être compatible ou satisfaire le scénario. On a les mêmes nuances que précédemment.

On distingue certains cas de sessions particulières où la reconnaissance est simplifiée. On parle de session datée lorsque tous les événements observés et reportés dans la session sont datés précisément. Une telle session peut en plus être ordonnée si l’ordre d’introduction incrémentale des événements correspond réellement à leur ordre chronologique. Enfin la reconnaissance s’effectue en temps courant s’il n’y a aucun retard entre l’occurrence des événements et leur introduction dans le système. Dans ce contexte on peut établir un agenda. Pour vérifier le scénario, l’événement suivant observé doit obligatoirement être dans l’agenda et vérifier les contraintes temporelles spécifiées par l'agenda.

B) Système IxTeT

a) Représentation des chroniques

Dousson et Ghallab [Dousson 94.1], [Dousson 94.2] ont créé un système de suivi et reconnaissance de chroniques fondé sur la propagation de contraintes temporelles numériques et l’association de fenêtres temporelles avec chaque événement. On dispose de modèles de chroniques ou de scenarii d’évolutions prévisibles que l’on cherche à reconnaître. Le système est prédictif au sens où il prédit tous les événements attendus par les hypothèses de chroniques en cours.

Ce système a déjà été utilisé dans deux applications dans le cadre de projets Esprit. Le premier, SKIDS, consistait en la surveillance du cycle d’un convoyeur dans un atelier. Le convoyeur devait faire un certain circuit pour prendre un objet dans une pièce et aller le déposer dans une autre. La reconnaissance se fait à partir d’un scénario décrivant le cycle normal de fonctionnement et de quelques scenarii d’alertes prévoyant la chute de l’objet en cours de route, l’absence de l’objet à prendre, etc. Le second projet, TIGER, est de plus grande envergure puisqu’il concerne la surveillance en ligne du fonctionnement d’une turbine à gaz APU (Auxiliary Power Unit). C’est un système de diagnostic de panne avec plusieurs scenarii de pannes plus ou moins complexes et qui ont nécessité une base d’expertise importante.

La modélisation des contraintes temporelles se fait à l'aide d'une version du logiciel IxTeT qui utilise des contraintes numériques (cf. II.2.3 p.37). Comme précédemment, un scénario est représenté par un graphe avec des contraintes temporelles. Les intervalles de temps sont exclusivement fermés, il n'y a plus la possibilité d'exclure certaines valeurs. Pour des raisons de complexité algorithmique, on essaye d'éviter les contraintes disjonctives. Soit on décompose le graphe disjonctif en graphes simples, soit on relâche les contraintes. Par exemple, si on a une contrainte $[1,5]\cup[8,10]$ on la transforme en $[1,10]$. Mais il faut être prudent avec ce genre de simplifications.

Par rapport au système précédent, on ajoute la notion d'assertion. Une assertion est la persistance d'un fait sur un certain intervalle de temps $[e_1,e_2]$. L'assertion est vérifiée si sur cet intervalle aucun événement ne vient contrarier le fait associé à l'assertion. Certains événements sont donc interdits pendant cette période, et on maintient en permanence une liste des événements interdits. $<P,e_1,e_2>$ signifie que l'événement P est interdit entre e_1 et e_2. Si on reconnaît P dans cet intervalle de temps, l'assertion est violée.

b) Calcul des fenêtres temporelles

Dans un premier temps, on effectue un prétraitement sur les données, on transforme tous les scenarii initiaux en leurs graphes minimaux complets(cf. graphe minimal de Fig. 30), comme dans la méthode précédente. L'algorithme de propagation est le même. On peut avoir une version incrémentale de l'algorithme où on propage les contraintes au fur et à mesure qu'elles sont données. Le résultat est identique si le graphe est consistant, mais en cas d'inconsistance il permet de rejeter un sous-ensemble de contraintes. Ceci peut aider à mettre au point les scenarii et à trouver les causes d'inconsistance. Mais les contraintes rejetées dépendent de l'ordre dans lequel on introduit les contraintes, donc le sous-ensemble de contraintes rejetées n'est pas forcément minimal.

Le premier graphe ainsi obtenu ne contient que des informations relatives entre nœuds, mais dès que l'on reconnaît un des événements de la chronique, celui-ci est daté (par un intervalle fermé ou une date simple) et on peut propager la date à tous les autres éléments (cf. instanciation du nœud 1 de Fig. 30). Soit i un événement qui doit arriver 10 à 20 minutes avant j ($D(j-i)=[10,20]$) ; on sait que i arrive dans une fenêtre temporelle $D(i)=[10h20,10h25]$, on peut donc en déduire $D(j)=D(i)\oplus D(j-i)=[10h30,10h45]$, où $\oplus$ est l'opération de composition de contraintes.

Un algorithme de propagation par le graphe minimal permet de calculer pour chaque événement une fenêtre admissible. Si $D(k)=D$, alors pour $i\neq k$, $D(i)=D(i)\cap(D\oplus D(i-k))$. Au fur et à mesure de la reconnaissance, le processus restreint les fenêtres admissibles puisque chaque événement reconnu est une précision supplémentaire sur les contraintes éléments (cf. instanciation du nœud 3 de Fig. 30). La propagation se fait de telle sorte qu'une fenêtre admissible contienne toutes les dates possibles pour son événement correspondant et seulement celles-ci. Un événement est reconnu s'il correspond au type d'activité attendue et si sa date d'occurrence est dans sa fenêtre admissible.

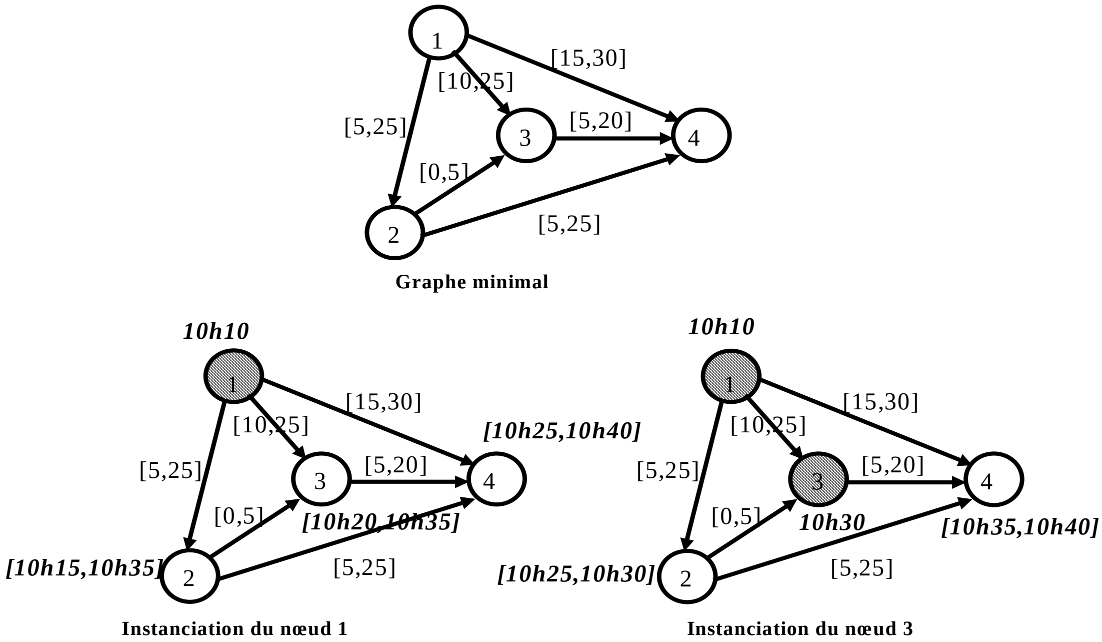


Fig. 30 : exemple de propagation de fenêtres temporelles

c) Reconnaissance

Le système de reconnaissance doit détecter tout ensemble d'événements qui correspond totalement ou partiellement à un modèle de chronique, en tenant compte des contraintes temporelles et des assertions. Tant que tous les éléments du modèle n'ont pas été détectés, on n'a qu'une instance de la chronique. La reconnaissance intervient lorsqu'il ne manque plus aucun élément du modèle. Le système maintient pour chaque événement attendu sa fenêtre admissible.

Une instance de chronique évolue par étapes issues des suivantes :

- i : création d'une instance vide à partir du modèle, elle attend encore tous les événements e_i et a toutes les assertions a_i à vérifier.
- ii : évolution de l'instance partielle par intégration d'un événement.
- iii : évolution de l'instance par suppression d'une assertion à vérifier.
- iv : mort et suppression de l'instance par violation d'une assertion.
- v : mort et suppression de l'instance par absence d'événement.
- vi : aboutissement de la reconnaissance de l'instance, tous les événements ont été reconnus et toutes les assertions vérifiées.

A l'aide des fenêtres temporelles, on peut définir des dates limites de validité d'une instance de chronique : la deadline et la holdline. La deadline est la plus petite borne supérieure des fenêtres admissibles des événements attendus. Passée cette date, si l'événement correspondant n'a pas été reconnu on peut abandonner la chronique : on est certain que cet événement ne pourra pas arriver en satisfaisant la contrainte (v). La holdline est la plus petite borne supérieure des fenêtres admissibles des événements de fin d'assertion. Passée cette date, si aucun événement ne pouvant violer l'assertion n'est arrivé, alors on peut considérer que l'assertion est vérifiée (iii).

Il y a propagation d'informations uniquement lors de la reconnaissance d'un nouvel événement ou lorsque l'on atteint une deadline ou une holdline. S'il n'y a pas d'événement, le système reste inactif jusqu'à sa prochaine date de réveil qui est la plus petite date limite de toutes les chroniques.

Lors de l'arrivée d'un nouvel événement, on compare dans un premier temps sa nature avec celle des différents éléments constituant les motifs de chroniques. S'il est apparié avec un de ces éléments, alors il est reconnu si et seulement si sa date d'occurrence est dans sa fenêtre admissible (ii). S'il vérifie toutes ces contraintes, alors on transforme sa fenêtre admissible en sa date d'occurrence et on propage cette nouvelle restriction temporelle au reste de la chronique pour effectuer la mise à jour des fenêtres admissibles.

Un événement peut aussi arrêter la persistance d'une assertion (iv). Soit une chronique S exigeant qu'aucun événement e n'intervienne entre e_1 et e_2 de fenêtres admissibles $D(e_1)$ et $D(e_2)$. Suivant les configurations de e_1 et e_2 et la date d'occurrence de e, on obtient les cas suivants :

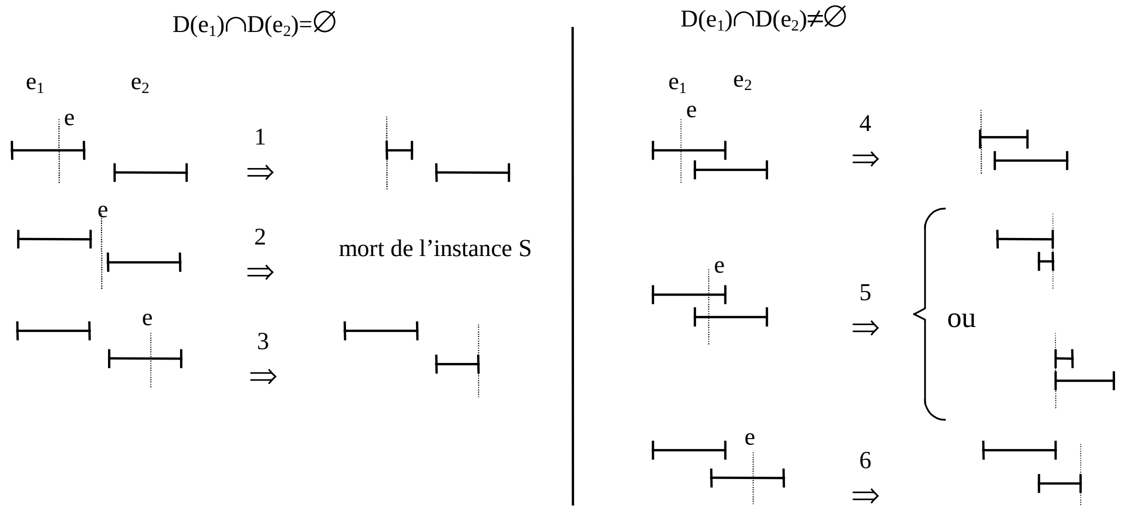


Fig. 31 : effet d'un événement interdisant la persistance d'une assertion

d) Retards

Entre l'occurrence d'un événement et sa perception, il peut y avoir un certain retard. Ce retard est variable et, par conséquent, l'ordre de reconnaissance des événements ne correspond pas forcément à l'ordre de leur occurrence. Ce retard est néanmoins borné. Pour un événement e, on note d(e) sa date d'occurrence, r(e) sa date de réception et $\Delta(e)$ l'intervalle contenant le retard. On a alors $r(e) \in d(e)+\Delta(e)$. Il faut tenir compte de ce retard pour le calcul des dates limites deadline et holdline.

De même pour une assertion a, le délai de traitement correspond au retard que l'on peut avoir avant d'être sûr qu'une assertion est vérifiée. Il correspond au retard possible du type d'événement pouvant violer cette assertion, on le notera $\Delta(a)$. En effet, passé ce délai, on est sûr qu'aucun événement, même retardé au maximum, ne pourra interrompre la persistance requise par l'assertion.

Ces retards impliquent aussi un délai sur la reconnaissance d'une chronique. Un événement terminal est un événement d'une chronique permettant potentiellement de conclure sur la reconnaissance de celle-ci. Ce n'est pas forcément une feuille du treillis puisqu'il faut tenir compte des retards potentiels. Les bornes du délai de la reconnaissance correspondent donc à l'union des retards possibles des événements terminaux.

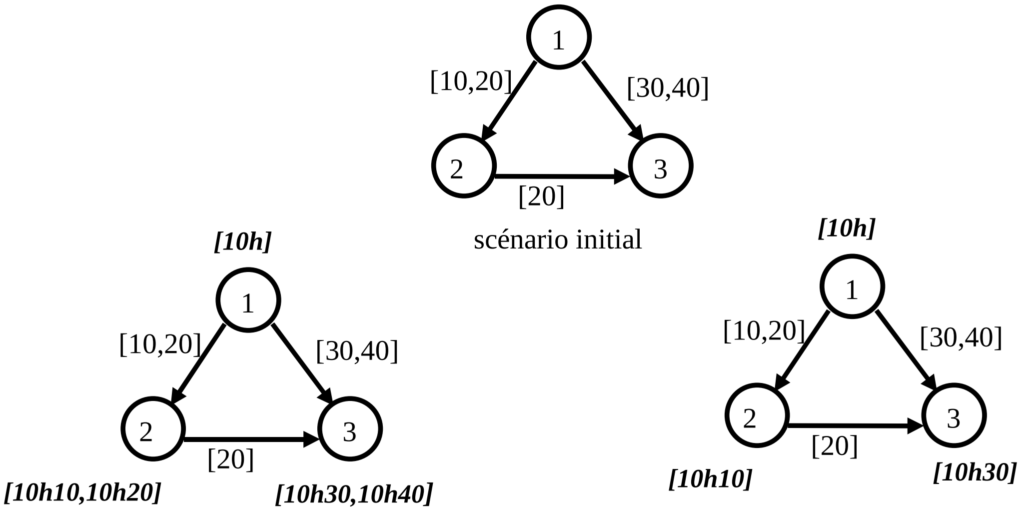
e) Gestion des instances de chroniques

Le système de reconnaissance doit faire face à un flot continu d'observations qu'il doit comparer à tous les scenarii disponibles dans sa bibliothèque. Lorsque un ou plusieurs scenarii sont instanciés, on attend des événements qui viendraient confirmer ou infirmer l'occurrence de ces scenarii. Il faut donc gérer et maintenir toutes les hypothèses de chroniques. On cherche à avoir une reconnaissance complète, c'est-à-dire qu'elle identifie toutes les instances de chroniques possibles parmi toutes les observations en entrée. Ceci nécessite de gérer à chaque instant un grand nombre d'hypothèses de chroniques.

De plus, on peut montrer que pour avoir une reconnaissance complète, à chaque fois qu'un événement est intégré à une chronique, il faut créer une nouvelle instance de cette chronique et garder l'ancienne telle quelle. Si on ne le fait pas, on risque de laisser passer un élément de reconnaissance comme le montre l'exemple suivant.

On considère un scénario initial simplifié. Ce scénario est consistant et on a son graphe minimal. Supposons que l'on ait les observations suivantes : 1 à 10h (événement du type 1 observé à 10h), 2 à 10h10, 2 à 10h15 et 3 à 10h35.

Voici ce qui se passe si on ne crée pas de nouvelle instance pour chaque événement :



scénario initial

Etape 1 : intégration de l'événement 1 à 10h

Etape 2 : intégration de l'événement 2 à 10h10

Fig. 32 : intégration d'événements sans création d'instances

Après l'étape 2, l'événement 2 n'est plus attendu donc on ne tient pas compte de l'arrivée de 2 à 10h15 et l'événement de type 3 arrive trop tard puisque sa fenêtre temporelle est réduite à [10h30]. Le scénario n'est pas reconnu alors que les événements 1 à 10h, 2 à 10h15 et 3 à 10h35 le vérifient parfaitement. Avant d'intégrer l'événement 2 à 10h10, il faut copier l'état de l'instance de chronique. Ceci permet ensuite d'intégrer les événements 2 à 10h15 puis 3 à 10h35 et donc de reconnaître la chronique.

On voit ici l'intérêt d'introduire un mécanisme de duplication pour garder toutes les formes possibles d'instances d'un même scénario. Toutes ces instances sont gérées globalement sous une forme arborescente (cf. annexe VII.2 p. 246). Le danger du mécanisme de duplication est d'entraîner une explosion combinatoire du nombre d'hypothèses. On peut la limiter en ajoutant des heuristiques particulières. On pourrait par exemple supprimer toutes les instances d'un arbre dès que l'on a abouti à la reconnaissance de la chronique associée à cet arbre. La plupart du temps en effet, plusieurs instances d'une même chronique aboutissant à une reconnaissance ne sont pas en recouvrement, donc on peut supprimer toutes les instances partielles en recouvrement avec l'instance reconnue (c'est l'arbre entier).

Un autre moyen de limiter le nombre d'instances serait de ne pas propager tout de suite un événement mais d'attendre et éventuellement de reprendre cet élément si nécessaire. Cela peut éviter de développer des instances qui de toute façon seraient mortes rapidement.

f) Imprécision et incertitude

Dousson [Dousson 94.1] envisage de gérer des événements approximativement datés avec un système similaire. Pour cela on peut modéliser les dates par des intervalles numériques sur lesquels on peut effectuer les opérations de somme, de différence, d'intersection... Les algorithmes de propagation sont alors inchangés.

Mais il faut revoir le mécanisme de reconnaissance d'un événement dans une chronique. La possibilité d'intégrer e dans une instance de chronique est :

$$\pi(e) = \frac{\lambda(d(e) \cap D(e))}{\lambda(d(e))}$$ où λ donne la longueur de l'intervalle.

L'ensemble I(e) des dates possibles de e dans S (intervalle propagé dans la chronique via les contraintes temporelles) est :

$$\pi(x, I(e)) = \begin{cases} \pi(e) \text{ si } x \in d(e) \cap D(e) \\ 0 \text{ sinon} \end{cases}$$

La propagation des dates reste inchangée (en remplaçant les opérateurs classiques par des opérateurs flous) mais nécessite une remise à jour des possibilités des événements déjà intégrés et dont la fenêtre admissible I(e) se trouve réduite de nouveau :

$$\begin{cases} I(e) \rightarrow I'(e) \subset I(e) \\ \pi'(e) = \pi(e) \times \dfrac{\lambda(I'(e))}{\lambda(I(e))} \end{cases}$$

Pour une assertion de date de début start(a) et de fin end(a), on évalue son degré de possibilité par :

$$\pi(a) = \min(\pi(start(a)), \pi(end(a)))$$

Le degré de possibilité d'une chronique complète est le minimum des possibilités de reconnaissance des événements et des assertions de cette chronique.

On pourrait aussi étendre le raisonnement à des événements incertains pour reproduire l'incertitude de mesure de capteurs par exemple. L'hypothèse simplificatrice de la complétude des observations et des modèles n'est pas toujours possible. Il faudrait pouvoir adapter le raisonnement au cas de l'incomplétude, en ajoutant des axiomes sur le domaine par exemple.

II.3.3 Comparaison avec la technique des gabarits

Le contexte opérationnel de la technique des gabarits est exactement le même que celui des XPlans ou des scripts. On doit faire face à un flot de données important, ces données étant imprécises, incertaines et incomplètes. Les observations sont perçues dans un ordre pas forcément chronologique, on a donc également la contrainte d'établir un traitement des informations qui soit indifférent de leur ordre d'arrivée. On peut adapter toutes les techniques de reconnaissance présentées pour vérifier cette condition.

Les gabarits vont en fait emprunter des éléments à plusieurs de ces mécanismes. La technique des gabarits présente en effet à la fois une partie reconnaissance structurelle, que l'on peut rapprocher principalement du formalisme de Kautz, et une partie reconnaissance temporelle tout aussi importante. Le temps joue effectivement un rôle primordial, et on peut voir la technique des gabarits d'une part comme un problème de satisfaction de contraintes temporelles et d'autre part comme un problème logique. Les deux aspects sont essentiels, ce qui distingue cette technique de toutes les autres pour lesquelles on a toujours un des aspects qui est prépondérant.

Une autre différence fondamentale vient de l'incomplétude des observations. On tient en effet compte du fait qu'un événement peut avoir lieu sans pour autant être observé, ce qui est lié au contexte. Ceci se traduit par la possibilité d'une reconnaissance sur des données incomplètes : la reconnaissance n'est pas mise en échec si un élément n'a pas été reconnu, on diminue simplement le degré de confiance de l'hypothèse associée.

A) Aspects logiques

D'un point de vue logique, la technique des gabarits se rapproche le plus fortement du formalisme de Kautz. La décomposition hiérarchique des plans est assez similaire. La relation de spécialisation correspond à une agrégation disjonctive et la relation de composition à une agrégation conjonctive. Les gabarits sont cependant un peu plus riches puisqu'ils proposent des formes d'agrégation plus complexes ainsi que des itérations. On ne trouve pas l'équivalent de la relation de permission de Hong, mais on peut en fait reproduire cette relation en jouant sur les éléments déclencheurs. L'axiome d'exclusion est lui l'équivalent de la notion de gabarit exclusif qui permet de rendre des hypothèses concurrentes.

Peut-on dire que la technique des gabarits relève d'un mécanisme déductif comme la reconnaissance selon Kautz ? On ne déduit pas de l'arbre de chaque gabarit composite un ensemble d'axiomes, mais on peut voir le mécanisme de reconnaissance structurelle comme la traduction de cet arbre sous forme de règles du type : agrégation(éléments fils)$\Rightarrow$élément père avec un certain degré de certitude. Suivant les observations appariées au niveau des nœuds fils, on déduit l'élément du nœud père avec un degré de confiance.

Mais ces déductions sont révisables. Le degré de confiance résultant peut en fait varier avec le temps puisque les degrés d'appariement évoluent. L'influence du temps est telle que l'on peut en fait considérer que les arbres constituent des bases de règles variables avec le temps. On peut donc faire le rapprochement avec une logique déductive non monotone.

La reconnaissance à base d'XPlans est différente puisqu'elle résulte d'un mécanisme abductif. On retrouve néanmoins les mêmes éléments représentés sous une autre forme, en dehors des préconditions et post-conditions qui n'apparaissent pas dans les gabarits. La notion de précondition pourrait être intéressante. On pourrait en effet associer un contexte à chaque gabarit et ne considérer ce gabarit que dans le cas où le contexte est vérifié. Le contexte pourrait être lié au type de terrain, au type d'armée, à la situation diplomatique du moment... Il agirait un peu comme un élément déclencheur de niveau supérieur sans toutefois entrer dans la composition même du gabarit comme une sous-action. Pour ce qui est de l'intérêt de post-conditions, il semble limité puisque l'on s'attache dans la technique des gabarits à prévoir ce qui va arriver et non pas à analyser ce qui s'est passé.

La décomposition hiérarchique des XPlans montre une différence essentielle par rapport aux gabarits. Cette décomposition n'utilise que des disjonctions, ce qui implique qu'un seul plan peut être reconnu alors qu'avec les gabarits plusieurs scenarii peuvent être reconnus simultanément. L'hypothèse que l'agent poursuit un but unique n'est plus valable.

Les seuls éléments logiques pris en compte par les gabarits sont des propositions représentant des événements. La notion d'assertion est absente. On pourrait l'introduire d'une façon similaire à ce qui est fait dans IxTeT, en créant une liste d'événements interdits sur certains intervalles de temps.

B) Aspects temporels

Les aspects temporels des gabarits peuvent aussi se retrouver dans les mécanismes de reconnaissance fondés sur la logique. Le mécanisme le plus éloigné est en fait la reconnaissance par automates à états finis. Le seul point commun avec les gabarits est la représentation de contraintes numériques de durées entre les événements. Les automates sont trop rigides dans leur représentation et limités en pratique à des scenarii très simples.

Ils ne permettent pas de traiter des contraintes symboliques et ne sont efficaces que pour des scenarii ordonnés et linéaires. Les scenarii modélisés par les gabarits sont loin de vérifier ces critères, ils sont beaucoup plus complexes. L'inconvénient des automates est qu'il faut décrire de façon explicite toutes les transitions possibles, ce qui laisse peu de marge de manœuvre.

Contrairement aux XPlans et aux scripts qui traitent les éléments temporels comme n'importe quel autre paramètre, le formalisme de Kautz accorde au temps un statut particulier. Comme pour les gabarits, un intervalle est la donnée de 4 réels <a,b,c,d>, où a et b sont les dates de début au plus tôt et au plus tard et c et d les dates de fin au plus tôt et au plus tard. Les seules relations temporelles sont les relations symboliques d'Allen. On pourrait d'ailleurs ajouter aisément aux gabarits ces relations en plus des relations numériques et des relations symboliques du type "le lendemain" ou "dans la nuit".

On retrouve chez Kautz comme chez Dousson ou chez Fontaine un mécanisme de propagation des contraintes similaire à celui utilisé dans les gabarits. De plus, l'hypothèse formulée par Song et Cohen selon laquelle une action débute avec sa première sous-action et se termine avec sa dernière sous-action est aussi vraie dans la technique des gabarits. Les gabarits sont plus riches que le formalisme de Kautz pour leur aspect numérique, avec entre autres la modélisation de durées et de contraintes relatives à ces durées.

Chez Wainer [Wainer 98], le temps intervient comme chez Kautz dans la deuxième phase de la reconnaissance après un raisonnement purement logique. Il utilise des contraintes temporelles floues, mais le raisonnement temporel employé oblige une représentation sous la forme d'un arbre. Ce modèle est donc plus restrictif que les gabarits pour l'expression des contraintes temporelles. On ne peut lier un symptôme qu'à celui qui le précède. D'autre part le modèle n'introduit pas les dates de fin des symptômes puisque dans une maladie on observe en général les débuts et les durées des symptômes. Dans les gabarits il se peut que l'on observe uniquement la fin d'une activité, il faut donc modéliser explicitement les dates de fin d'activités. Une modélisation des contraintes temporelles par un graphe semble donc beaucoup plus adaptée.

La reconnaissance par comparaison de graphes introduit la notion de scénario actualisé. C'est le graphe où les contraintes introduites par les observations ont été propagées. Le scénario actualisé fait apparaître directement les fenêtres temporelles des événements : elles correspondent aux contraintes entre le nœud représentant l'origine des temps et le nœud de l'événement considéré. La technique des gabarits utilise le même raisonnement mais le calcul des fenêtres temporelles est fait de façon ad hoc sans méthode particulière.

Le raisonnement est un peu différent avec IxTeT. On effectue aussi une mise à jour des fenêtres temporelles mais les contraintes ne sont pas actualisées. De ce fait, IxTeT ne permet pas d'exprimer des observations sur des durées ou des délais, il n'admet que des événements datés et ne peut pas prendre en compte une information comme : "l'activité A a été observée hier pendant 1 heure". Il ne fait pas apparaître les contraintes actualisées qui sont une information supplémentaire par rapport aux fenêtres temporelles et constituent des prédictions sur les délais et les durées à respecter.

Les notions d'agenda et de date du prochain événement au plus tard sont intéressantes. S'il n'y a pas d'assertion, on ne peut pas définir de holdline mais en revanche on peut utiliser une deadline. La notion d'agenda ne peut pas s'appliquer au cas de sessions incomplètes. On peut en effet manquer l'observation d'un élément de l'agenda si l'observation est incomplète et avoir ensuite un élément qui pourrait être apparié mais qui n'est pas dans l'agenda faute de mise à jour après l'observation manquée.

Dans la technique des gabarits, les temps d'obsolescence et de validité remplacent la notion d'agenda. Ces temps ne sont pas associés à un événement particulier attendu mais au gabarit lui-même et permettent d'éviter de garder indéfiniment des hypothèses qui n'ont plus aucune chance d'évoluer ou dont la certitude est devenue trop faible. Mais à titre d'information on pourrait ajouter un agenda des événements et de leurs dates d'occurrence prévues. L'agenda ne jouerait aucun rôle dans la gestion des hypothèses mais serait une information supplémentaire pour l'officier de renseignements.

C) Gestion et évaluation des hypothèses

La manière d'évaluer les différentes hypothèses de scenarii en cours de reconnaissance est assez variable et ne dépend pas du choix entre reconnaissance structurelle et temporelle. Chez Kautz par exemple, il n'y a pas d'évaluation numérique des hypothèses. On utilise une heuristique de minimisation du nombre d'hypothèses justifiant tous les événements observés. On pourrait y ajouter comme chez Wainer un degré de consistance temporelle global. Mais alors on sépare totalement les aspects structurels et temporels. De même la technique des graphes donne une certaine idée de gradualité dans la reconnaissance (symbolique ou numérique) mais les degrés d'évaluation des hypothèses ne prennent en compte que l'aspect temporel.

Dans les gabarits on souhaite ajouter à la consistance temporelle des critères liés à la structure comme l'importance et le nombre de sous-activités reconnues. Plus un gabarit intègre d'observations, plus son degré de confiance a de chance d'être grand. Toutes les observations n'ont pas la même importance, ce qui est intuitivement tout à fait justifié, surtout dans le cadre d'une reconnaissance incomplète. Le degré de reconnaissance d'un gabarit augmente donc aussi avec l'importance des sous-activités reconnues. Les XPlans et les scripts utilisent eux aussi des degrés de confiance mais leur calcul est plus arbitraire et il n'y a pas cette notion d'importance d'un événement.

Un degré de compatibilité temporelle global n'est pas satisfaisant, il vaut mieux introduire la notion de compatibilité partielle entre une observation et un élément de scénario, comme le propose Dousson dans son introduction de degrés de possibilité. Un événement peut en effet être imprécisément daté et correspondre de façon incertaine à un élément attendu du scénario, et la compatibilité partielle permet de ne plus le rejeter comme auparavant. L'évaluation globale pourra ainsi tenir compte à la fois de la compatibilité partielle de chacun des éléments reconnus, de leur importance et des nombres d'éléments reconnus et non reconnus.

La gestion des hypothèses dans IxTeT fait apparaître nettement le problème d'une reconnaissance qui se veut complète. La gestion des hypothèses est encore plus cruciale avec des degrés de compatibilité partielle. Dans ce seul cas en effet, un nouvel appariement peut diminuer la confiance dans une instance en dégradant la satisfaction des contraintes temporelles. Il existe alors au moins une instance partielle extraite de cette instance complète plus certaine que cette dernière. Ce qui signifie que l'on devrait garder en permanence toutes les instances partielles possibles d'une hypothèse pour être sûr de développer à chaque instant l'instance correspondant à la meilleure reconnaissance du gabarit. Le mécanisme de duplication le permet, mais à quel prix ? De plus, les observations dans les gabarits étant incomplètes, une instance ne meurt plus lorsqu'elle atteint sa deadline ; seul son degré de confiance diminue. Une instance ne meurt que lorsqu'elle dépasse le temps d'obsolescence.

Si on reproduit le mécanisme de duplication propre à IxTeT dans les gabarits, on risque donc d'arriver à une explosion du nombre d'instances en cours. Il faut absolument les limiter par des heuristiques. L'utilisation d'un seuil de déclenchement (comme pour les scripts) et d'éléments déclencheurs peut faire partie de ces heuristiques. On ne considère les gabarits que lorsqu'ils possèdent assez d'éléments caractéristiques. Le temps de validité participe aussi à la même idée en éliminant plus tôt des instances.

Une autre idée serait de ne pas utiliser le mécanisme de duplication, de ne développer qu'une seule instance de gabarit. Dans ce cas il faut mettre au point des techniques de retour arrière pour avoir effectivement une reconnaissance complète. Actuellement, les gabarits fonctionnent déjà avec des retours arrière mais il faut adapter ces techniques au cas délicat de la compatibilité partielle. On peut s'inspirer de ce qui est proposé avec les automates à états finis, où dans un premier temps on ne regarde que les contraintes structurelles, et ensuite seulement on vérifie les contraintes temporelles.

II.4 Conclusion

La particularité de la technique des gabarits tient à la fois dans son appartenance à un formalisme logique proche de celui de Kautz et dans l'importance de son aspect temporel. C'est ce qui fait sa richesse. On souhaite l'enrichir encore en généralisant les mécanismes de propagation à tous types de contraintes (symboliques, floues) et en autorisant les contraintes entre tout couple de nœuds. Des contraintes floues permettraient d'avoir une plus grande finesse dans la reconnaissance. La notion de satisfaction partielle d'une contrainte est primordiale pour l'évaluation de la reconnaissance d'un modèle de scénario.

Il faut absolument avoir un degré de compatibilité partielle au niveau de chaque élément et pas globalement comme dans la méthode de comparaison de graphes. On peut ainsi tenir compte de la structure des éléments et de sa signification dans l'évaluation de l'hypothèse. Le respect de la structure et des contraintes temporelles sont évalués simultanément. Ceci permettra par ailleurs de bien prendre en compte l'incomplétude des données. L'absence d'un événement sera prise en compte par une compatibilité partielle nulle de l'événement considéré alors qu'avec un degré de compatibilité globale cette absence n'aurait sans doute pas influencé l'évaluation.

Le système de Dousson n'est pas applicable à notre cas puisqu'il ne permet pas d'introduire des informations directement sur des contraintes observées, mais seulement sur des événements datés. Mais on peut s'inspirer de son idée de calcul du degré de compatibilité partielle. Au lieu de comparer les fenêtres temporelles théoriques et réelles d'un événement, on comparera plutôt des contraintes théoriques et réelles. Pour cela il faut disposer des contraintes théoriques mises à jour, ce que permet le graphe du scénario actualisé mais pas IxTeT.

Il semble intéressant de dissocier dans la représentation d’une part l’aspect structurel en gardant la modélisation hiérarchique et d’autre part l’aspect temporel en traduisant chaque arbre par un graphe temporel permettant des contraintes temporelles plus riches. L'avantage des graphes temporels est d'offrir une méthode formelle et générale pour la propagation des contraintes. On pourrait travailler sur le graphe du scénario actualisé pour obtenir les fenêtres temporelles et pour tous les calculs de compatibilité. Nous allons voir en détail dans le chapitre suivant toutes les possibilités offertes par une telle représentation et les extensions possibles à la logique floue.

Un problème important reste encore à résoudre : celui de la gestion des hypothèses dans le cadre d’une reconnaissance complète avec des observations incomplètes et de la compatibilité partielle. Nous avons évoqué le choix entre heuristiques et retours arrières, mais la solution est peut-être dans une combinaison des deux.

III Reconnaissance par comparaison de graphes temporels flous

Le chapitre précédent fait apparaître que la technique des gabarits gagnerait à adopter une représentation temporelle sous la forme d'un graphe de contraintes. Une telle représentation permet en effet une grande richesse dans le pouvoir d'expression et le graphe du scénario actualisé met à jour les fenêtres temporelles ainsi que les contraintes liées à chaque élément du graphe. Ainsi, on pourra comparer le graphe du scénario actualisé et le graphe de la session, non pas globalement, mais élément par élément, afin d'intégrer dans l'évaluation de la reconnaissance une compatibilité temporelle de chaque élément. L'évaluation tiendra compte simultanément de la structure et de la cohérence temporelle.

On souhaite aussi introduire des contraintes floues pour mieux modéliser l'imprécision et pour exprimer que des contraintes temporelles peuvent être partiellement satisfaites [Eude 97], [Eude 98.2]. On entre donc dans le cadre des FCSP, extension des CSP à des contraintes floues. Nous allons voir dans un premier temps quels sont les mécanismes de raisonnement liés aux FCSP. Nous aborderons ensuite le passage du gabarit à un graphe de contraintes temporelles floues, graphe sur lequel nous nous appuierons pour toute la partie propagation des contraintes et obtention du scénario actualisé. Nous conclurons sur le pouvoir d'expression temporelle d'une telle représentation suivant la modélisation choisie pour les ensembles flous.

III.1 CSP flous

De nombreux problèmes de l’intelligence artificielle sont résolus par une approche orientée vers les problèmes de satisfaction de contraintes. Mais les contraintes exprimées sont alors très rigides : une solution satisfait totalement ou pas du tout une contrainte et les contraintes sont certaines. Dans la réalité, de nombreux problèmes exprimés ainsi s'avèrent insolubles et il est nécessaire de relâcher certaines contraintes. Certains auteurs ont donc essayé d’étendre le formalisme CSP à des contraintes exprimées de façon plus souple par la théorie des sous-ensembles flous ou par la théorie des possibilités.

Fargier [Fargier 94] s'est intéressée la première aux problèmes de satisfaction de contraintes floues (FCSP). Elle a étendu aux FCSP un bon nombre des notions relatives aux CSP. Certaines de ces notions ont été appliquées dans le cadre de problèmes d'ordonnancement de production. Nous allons donc voir comment on peut raisonner sur des graphes temporels flous à partir de ces travaux et d'autres qui sont complémentaires. Le raisonnement peut porter sur l'évaluation du degré de satisfaction d'un FCSP, sur son niveau de consistance, sur la propagation des contraintes ou sur la recherche de solutions. Nous allons détailler tous ces points.

III.1.1 Définition et interprétation des contraintes floues

Les contraintes classiques sont précises et s'expriment par une condition de type booléenne qui peut être totalement vérifiée ou pas du tout, mais il n'y a pas de position intermédiaire. Toutes les contraintes ont la même valeur, il n'y a pas de hiérarchie entre elles. Nous allons voir les différentes manières d'élargir ces formes de contraintes pour une plus grande souplesse.

A) Les contraintes souples

La première amélioration consiste à pouvoir exprimer qu'une contrainte est partiellement satisfaite. On veut pouvoir dire qu'elle n'est pas parfaitement satisfaite mais qu'elle l'est avec un niveau suffisant pour considérer qu'elle n'est pas violée. On utilise alors une relation floue [Dubois 95], [Fargier 94] et [Fortemps 95]. Une contrainte C s'appliquant sur un vecteur d est modélisée par une fonction d'appartenance μ_R que l'on interprète par : $\mu_R(d)=1$ si d satisfait totalement C, $\mu_R(d)=0$ si d ne satisfait pas du tout C et $0<\mu_R(d)<1$ signifie que d satisfait partiellement C. Plus $\mu_R(d)$ est proche de 1, plus la contrainte est satisfaite.

B) Les contraintes à priorité

On peut aussi souhaiter exprimer qu'une contrainte est plus importante qu'une autre, au sens où, si on ne peut pas satisfaire les contraintes simultanément, une solution qui satisfait la contrainte la plus importante est meilleure qu'une autre qui ne satisferait qu'une contrainte moins importante. On affecte un coefficient de priorité Pr(C) compris entre 0 et 1 à chaque contrainte C. Pr(C) peut être vu comme le degré de nécessité de satisfaire C, et son complémentaire 1-Pr(C) comme le degré de possibilité de violer C [Dubois 95], [Fargier 94].

Si on reprend la modélisation précédente avec une relation floue μ_R, on peut écrire $\mu_R(d)=1$ si d satisfait C et $\mu_R(d)=1-Pr(C)$ sinon. Même si C est violée, si elle est peu importante elle sera considérée comme pratiquement satisfaite. En fait, plus son degré de priorité est faible, plus son influence est réduite. Si une contrainte est à la fois flexible et a un degré de priorité, on fusionne les deux informations en une nouvelle relation floue $\mu_{R'}(d)=\max((1-Pr(c),\mu_R(d))$.

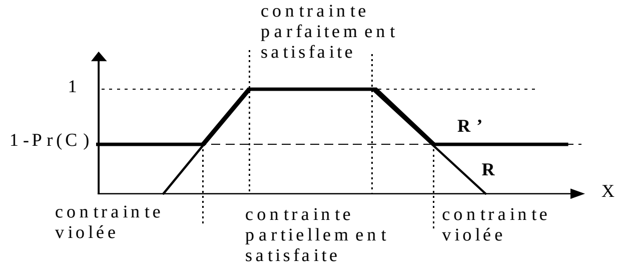


Fig. 33 : exemple de contrainte flexible à priorité

C) Les contraintes avec garde-fou

Certaines contraintes ne peuvent être relaxées que jusqu'à un certain point, il faut fixer les limites à ne pas dépasser. Soit par exemple une contrainte floue C représentée par μ_R et de priorité Pr(C). On peut exprimer que C est une contrainte plus forte que C' représentée par $\mu_{R'}$ en établissant que C ne peut être relaxée au-delà de C'.

La relation floue qui en résulte vaut : $\mu_{R*}(d)=\min[\max(1-Pr(C),\mu_R(d)),\mu_{R'}(d)]$ [Fargier 94], [Dubois 95]. En voici une représentation :

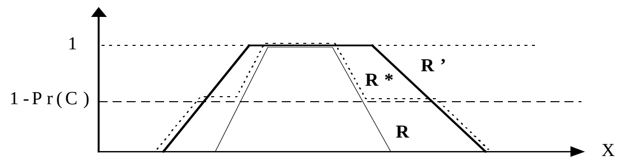


Fig. 34 : exemple de contrainte avec garde-fou

D) Les contraintes conditionnelles

Dans certains problèmes, lorsque des conditions essentielles ne sont pas vérifiées, il n'est même pas besoin de regarder d'autres conditions qui leur sont liées. Une contrainte C_j peut ainsi être conditionnée par une autre contrainte C_i. Si C_i n'est pas satisfaite, alors il n'est pas utile de satisfaire C_j et son degré de priorité devient nul, alors que si C_i est satisfaite il faut considérer C_j et lui affecter un degré de priorité de 1. Dans le cas flou, le degré de priorité de C_j est le degré de satisfaction de C_i : $\mu_{R(C_j)}(d)=\max(\mu_{R_j}(d),1-\mu_{R_i}(d))$ [Fargier 94], [Dubois 95].

E) Les contraintes incertaines

On distingue deux types d'incertitude dans la définition de contraintes. Des contraintes peuvent être incertaines parce qu'elles touchent des paramètres incontrôlables. On peut par exemple avoir une contrainte sur la durée écoulée entre deux événements, cette contrainte étant impérative mais la durée étant imparfaitement connue. La contrainte est donc mal connue, incertaine.

On peut aussi avoir des contraintes dont l'existence même est incertaine. Le problème réel comporte de façon sûre un ensemble de contraintes C, mais parmi un ensemble de contraintes C', on ne sait pas lesquelles figureront dans le problème. Cela dépend du déroulement de l'action. Ce sont des contraintes dont on ne connaît pas le degré de priorité à l'avance ; il peut être nul (inexistence de la contrainte), ou prendre toute autre valeur. On peut en fait ramener toutes les contraintes incertaines à des contraintes souples à priorité [Fargier 94].

III.1.2 Evaluation des FCSP

Les contraintes souples permettent d'exprimer qu'une solution satisfait plus ou moins une contrainte. Pour un problème constitué de plusieurs contraintes, une solution satisfait plus ou moins chacune de ces contraintes et pour évaluer la solution globalement ou pour comparer plusieurs solutions il faut tenir compte de chacun des degrés de satisfaction partielle. La solution satisfait le problème avec un degré de satisfaction global qui peut être obtenu par différentes méthodes, la plus courante étant la méthode égalitariste.

A) Vision égalitariste

Pour une instanciation donnée d, il s'agit d'évaluer dans quelle mesure elle peut être solution du problème. On considère que le problème est constitué uniquement de contraintes flexibles modélisées par une relation floue puisque toutes les contraintes envisagées peuvent se traduire ainsi, les intervalles étant un cas particulier de relation floue.

On peut donc évaluer la satisfaction de chaque contrainte C_i par $\mu_{R_i}(d)$. S'il existe un indice j tel que $\mu_{R_j}(d)=0$, alors la contrainte C_j est violée et d n'est pas solution du problème. Sinon, le point de vue égalitariste évalue la satisfaction d'un problème par le degré de satisfaction de la moins satisfaite des contraintes [Fargier 94]. Le degré de satisfaction global d'une instanciation est :

$$sat(d)=\min_{i=1\ldots n}\mu_{R_i}(d)$$

Par conséquent, la meilleure solution au problème est celle qui maximise le degré de satisfaction de sa contrainte la moins satisfaite. Le degré de satisfaction de la meilleure solution est aussi le degré de consistance du problème P :

$$cons(P)=\sup_{d}\,[\min_{i=1\ldots n}\mu_{R_i}(d)]$$

L'intérêt de cette évaluation égalitariste est qu'il n'y a pas d'effet de compensation : une contrainte parfaitement satisfaite ne peut pas compenser une contrainte très faiblement satisfaite, ce qui assure un minimum d'homogénéité de la solution. Ceci est valable dans certains types de problèmes comme en ordonnancement flexible où l'on recherche un maximum de sécurité sur toutes les contraintes.

a) **E**ffet de noyade

La méthode égalitariste comporte néanmoins un inconvénient : plusieurs solutions peuvent avoir le même degré de satisfaction alors qu'intuitivement certaines sont plus satisfaisantes que d'autres [Fargier 94]. En fait, on réduit toute l'information sur la qualité de la solution à la contrainte la moins satisfaite, ce qui masque un certain nombre de renseignements complémentaires, d'où le terme "effet de noyade".

Par exemple, si on compare deux solutions d_1 et d_2 telles que : d_1 satisfait toutes les contraintes sauf une avec un degré de 1, sa contrainte la moins satisfaite l'étant avec un degré a, et d_2 satisfait toutes les contraintes avec un degré a. d_1 et d_2 ont le même degré de satisfaction global a, mais intuitivement, il est incontestable que d_1 est une meilleure solution que d_2. Suivant le type de problème, ces considérations peuvent n'avoir aucune importance. Sinon, on peut raffiner le modèle en utilisant d'autres types de comparateurs comme l'ordre lexicographique par exemple.

b) Ordre lexicographique

L'ordre lexicographique est le plus couramment utilisé pour raffiner l'ordre égalitariste. Il est fondé sur une représentation particulière de chaque solution sous la forme d'un ensemble flou défini sur l'univers discret des contraintes. On lui associe pour chaque contrainte le degré de satisfaction correspondant. Une fois cette représentation obtenue, il s'agit en fait de comparer des ensembles flous. La comparaison effectuée avec un ordre lexicographique est fondée sur des critères de cardinalité.

Définition 1 : u>v au sens lexicographique si, à l'α-coupe la plus basse qui les différencie, u comporte plus de contraintes satisfaites que v. On rappelle q'une contrainte C est satisfaite à l'α-coupe α ssi le degré de satisfaction de Cest strictement supérieur à α.

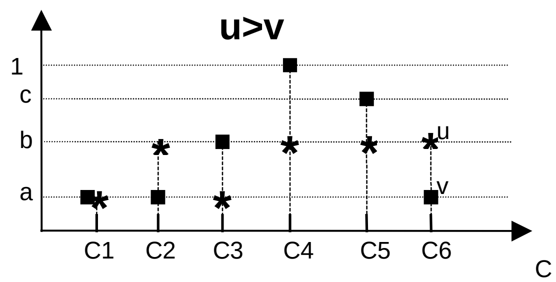


Fig. 35 : comparaison lexicographique

Sur la figure ci-dessus, à l'α-coupe a, u comporte 4 contraintes satisfaites (C2, C4, C5 et C6)alors que v n'en a que 3 (C3, C4 et C5). Au sens égalitariste, u et v sont deux solutions équivalentes de degré de satisfaction a alors qu'au sens lexicographique, u est une meilleure solution que v.

La vision égalitariste est malgré l'effet de noyade la plus courante et toutes les définitions relatives à la consistance locale reposent sur cette méthode de comparaison.

III.1.3 Notions de consistance locale

On peut étendre la majorité des définitions et des théorèmes relatifs aux CSP (cf. II.2.1B) p.31) au cas de contraintes flexibles [Fargier 94]. C'est le cas par exemple des définitions de consistance locale. Soit une instance partielle de solution $(d_1,\ldots,d_k)$, on évalue sa consistance partielle en combinant les satisfactions des relations floues.

Définition 2 : La consistance partielle de $(d_1,\ldots,d_k)$ est : $\text{cons}(d_1,\ldots,d_k) = \mu_{\left[\bigcap_i \left\{R_i \in R / V(R_i) \subseteq \{X_1,\ldots,X_k\}\right\}\right]}(d_1,\ldots,d_k)$ où R est l'ensemble des relations floues du problème, $V(R_i)$ est le domaine de définition de R_i et X_i la variable instanciée par d_i.

Chez Fargier, on considère qu'un graphe flou est consistant dès qu'il existe une solution dont le degré de satisfaction est strictement positif, alors que pour Marin [Marin 94] il doit être de 1. Les définitions de consistance locale reposent sur la consistance partielle d'une solution. En général, une propriété de consistance locale s'exprimera de la façon suivante : toute instanciation de a variables ou contraintes peut être étendue à une instanciation de b variables ou contraintes en gardant le même degré de consistance partielle. Comme dans le cas classique, on parle de k-consistance.

Définitions (3) :

- **k-consistance** : $\forall\{X_{j_1}\ldots X_{j_{k-1}}\}\in X$, $\forall$ $X_{j_k}\in X$ / $X_{j_k}\notin\{X_{j_1}\ldots X_{j_{k-1}}\}$, $\forall\{d_{j_1}\ldots d_{j_{k-1}}\}\in D_{j_1}\times\ldots\times D_{j_{k-1}}$, $\exists d_{j_k}\in D_{j_k}/\text{cons}(d_{j_1}\ldots d_{j_{k-1}},d_{j_k})=\text{cons}(d_{j_1}\ldots d_{j_{k-1}})$.

- **k-consistance forte** : un FCSP est fortement k-consistant ssi pour tout $i \leq k$, il est i-consistant.

- **2-consistance** : la définition de la 2-consistance s'obtient directement à partir de celle de la k-consistance et peut aussi se traduire par : pour toute contrainte C_{ij}, $R_i \subseteq [R_{ij} \cap R_j]^{\downarrow\{X_i\}}$ où ${}^{\downarrow\{Xi\}}$ signifie la projection sur le domaine de définition X_i. On obtient **l'arc-consistance** en ajoutant que toute relation R_i doit être non vide.

- **chemins-consistance** : un FCSP est dit chemins-consistant si, pour tout couple de variables, tout couple de valeurs localement consistant l'est au moins dans la même mesure par tous les chemins de contraintes liant ces deux variables.

On peut aussi étendre la notion de graphe minimal au cas flou. Mais si le degré de satisfaction global de la meilleure des solutions est compris strictement entre 0 et 1, avec Fargier on obtient un graphe minimal consistant alors que Marin déclare le problème inconsistant. On peut généraliser les théorèmes de Montanari (cf. II.2.1B)d) p.33).

De plus, pour des contraintes floues non disjonctives la composition est distributive sur l'intersection. On le montre en se ramenant à un calcul sur les α-coupes. Pour des contraintes floues non disjonctives, les α-coupes sont des intervalles fermés qui vérifient donc bien la distributivité. Avec les deux théorèmes de Montanari et la propriété de distributivité, on déduit que pour un graphe de contraintes numériques floues non disjonctives, la 3-consistance assure la minimalité.

III.1.4 Propagation des contraintes

De même que l'on a adapté les notions de consistance locale et de graphe minimal au cas flou, on peut généraliser aux FCSP les algorithmes de filtrage propres aux CSP [Fargier 94]. Pour représenter un ensemble flou, on utilise souvent ses α-coupes. Le nombre p d'α-coupes nécessaires pour modéliser l'ensemble flou est significatif de la finesse de la représentation. La complexité des algorithmes est multipliée par p. Plus p est grand, plus la représentation est fine mais aussi plus les calculs sont complexes.

A) Approches de type CSP

Comme nous l'avons vu précédemment, le calcul du graphe minimal avec des contraintes numériques classiques peut se faire par deux types d'approches : une approche du type CSP qui considère un graphe de contraintes et utilise des algorithmes dérivés de l'algorithme de Waltz (cf. II.2.1C)a) p.33), et une approche de type PERT qui considère un graphe de distances et utilise des algorithmes proches de l'algorithme de Floyd-Warshall (cf. II.2.2 p.36). Nous allons voir ici comment on peut adapter les algorithmes du type Waltz à des contraintes floues.

Fargier est le premier auteur [Fargier 94] à avoir généralisé les notions liées aux CSP à des graphes de contraintes floues. Elle a à ce titre démontré l'existence du graphe minimal dans le cas flou. PC1 et PC2 sont généralisés en remplaçant les opérations de composition et d'intersection respectivement par la somme et l'intersection floues. La composition des contraintes se ramène à une somme floue dans le cas de relations floues réduites à des nombres flous, ce que choisit Fargier et qui est beaucoup plus simple en pratique. Sinon il faut utiliser la composition classique de relations floues obtenue par l'opérateur "sup-min".

Le degré de consistance global du graphe minimal est le degré de consistance de la contrainte la moins satisfaite. Le degré de satisfaction d'une contrainte est son degré d'appartenance le plus fort, c'est-à-dire la hauteur de son ensemble flou. L'algorithme s'arrête dès que les modifications ne sont plus significatives ou si le degré de consistance global devient trop faible, auquel cas le graphe est déclaré inconsistant.

Marin [Marin 94] utilise lui aussi ce type d'algorithme, mais avec des contraintes qui sont exclusivement des contraintes normalisées. En fait, il considère que le graphe est inconsistant dès qu'une contrainte devient non normalisée. Ce choix simplifie certains calculs mais perd quasiment la notion de satisfaction partielle d'une contrainte.

L’intérêt d’utiliser des contraintes floues dans la technique des gabarits réside essentiellement dans la possibilité d’accepter des solutions qui ne satisfont pas totalement le graphe. Or chez Marin ce type de solutions est rejeté puisque seuls les graphes où toutes les contraintes sont normalisées sont consistants. Ainsi on trouve les mêmes solutions qu’en prenant comme contraintes les noyaux des ensembles flous, ce qui revient à prendre des intervalles numériques classiques comme contraintes.

En fait Marin n’utilise pas le graphe minimal dans l’optique de la recherche d’une solution. C’est simplement un moyen de traduire un scénario décrit de façon imprécise en langage naturel. Le graphe minimal est une forme plus précise du scénario décrit, où seules les informations cohérentes sont propagées afin de découvrir éventuellement de nouvelles relations entre événements. Ce n’est pas ce qui nous intéresse dans la technique des gabarits, où les solutions partiellement satisfaisantes sont importantes et doivent être gardées ; il est donc impératif de prendre en compte les contraintes non normalisées.

Dubois [Dubois 89] réalise lui aussi la propagation de contraintes floues dans des graphes. Contrairement aux deux auteurs précédemment cités, Dubois modélise ses contraintes par des relations floues. L’opération de composition est alors une composition floue "sup-min". Mais cette opération est beaucoup plus complexe à réaliser qu’une somme. Dubois autorise les ensembles flous non normalisés. Voici le principe de la propagation.

Soient $t_1 ... t_n$ des événements ou des durées floues contenus dans les ensembles flous $A_1 ... A_n$ et R une relation contraignant $A_1 ... A_n$. Les ensembles A_i peuvent donc être définis de façon plus fine par $A'_1 ... A'_n$ avec : $A'_i = \{t_i \in A_i \ / \ \forall\ j \neq i, \exists t_j \in A_j, (t_1 ... t_n) \in R\}$ avec :

$$\mu_{A'_i}(t_i) = \sup_{t_j, j\neq i} \min(\mu_R(t_1, \dots, t_n), \min_j \mu_{A_j}(t_j))$$

$A'_i = A_i \cap (R \circ \underset{j\neq i}{X} A_j)$ où $\circ$ est la composition sup-min et X le produit cartésien.

S’il y a plusieurs relations, on répète l’opération pour chaque relation jusqu’à ce qu’il n’y ait plus de changement. L’algorithme se poursuit tant que les contraintes sont non vides mais on ne sait pas s’il se termine dans le cas continu. On peut décider de l’arrêter lorsqu’il atteint un état assez stable ou lorsque le niveau de consistance du problème (minimum des hauteurs des contraintes) devient trop faible [Fargier 94].

B) Approches de type PERT

Les problèmes d’ordonnancement sont très proches des CSP puisque l’on peut les traduire par des contraintes de précédence entre tâches et de durées des tâches. Il y a des équivalences de l’un à l’autre, et [Dechter 91] a montré que les algorithmes du type Waltz pour les CSP sont équivalents aux algorithmes de type Floyd-Warshall pour le graphe de distances équivalent au CSP.

Lorsqu'un graphe temporel comporte une étiquette floue A_{ij} de distribution $\mu_{A_{ij}}$ (cas où A_{ij} est un nombre flou), cela signifie que X_j-$X_i \in A_{ij}$ ou X_j-$X_i \leq A_{ij}$ et X_i-$X_j \leq -A_{ij}$, $-A_{ij}$ étant un nombre flou avec $\mu_{-A_{ij}}$ telle que $\mu_{-A_{ij}}(x)=\mu_{A_{ij}}(-x)$. Ceci est équivalent à deux contraintes de distance d_{ij}=avant(A_{ij}) et d_{ji}=avant($-A_{ij}$)=-apres(A_{ij}). On peut donc facilement construire un PERT avec des distances floues équivalent à un graphe de contraintes floues lorsque les contraintes sont des nombres flous. De même pour revenir à un graphe de contraintes à partir de distances on effectue l'opération inverse A_{ij}=apres($-d_{ji}$)$\cap$avant(d_{ij}).

Il n'existe pas à notre connaissance de version floue de l'algorithme de Floyd-Warshall, mais en revanche il existe des versions floues de l'algorithme de Ford qui sert à déterminer dans les PERT les dates de début au plus tôt et de fin au plus tard des tâches. Ces deux algorithmes sont très proches et utilisent les mêmes types d'opérations. On pourrait de la même façon généraliser l'algorithme de Floyd-Warshall au cas flou en appliquant le principe d'extension, et donc en utilisant les opérateurs de minimum et de somme flous.

Chanas et Gazdik [Chanas 81] [Gazdik 83] ont appliqué l'algorithme de Ford flou en modélisant les distances floues par des α-coupes. Les opérations classiques de minimum et de somme floues peuvent ainsi être décomposées en opérations classiques sur les intervalles correspondant à chaque α-coupe.

Dubois et Prade [Dubois 78] appliquent le même algorithme avec des ensembles flous de type S-D définis par: S et D sont deux fonctions paires valant 0 aux infinis et 1 en 0. m, α et β sont les paramètres caractéristiques de A de fonction d'appartenance μ_A :

$$\mu_A(x)=S[(x-m)/\alpha] \text{ sur }]-\infty,m] \text{ et } \mu_A(x)=D[(x-m)/\beta] \text{ sur }]m,+\infty]$$

Les triangles font partie de cette famille d'ensembles flous. Somme et différence floues sont obtenues de façon classique par combinaison des paramètres caractéristiques, mais pour le minimum flou on est obligé d'approcher le résultat qui dans certains cas ne donne pas un ensemble flou de type S-D. Soient M=(m,a,b) et N=(n,c,d) deux nombres flous S-D.

Si m<n on a : min(M,N)=(m,a,b) et si m≅n on a min(M,N)=(min(m,n),max(a,c),min(b,d)). Cette approximation conserve le noyau du résultat mais pas son support.

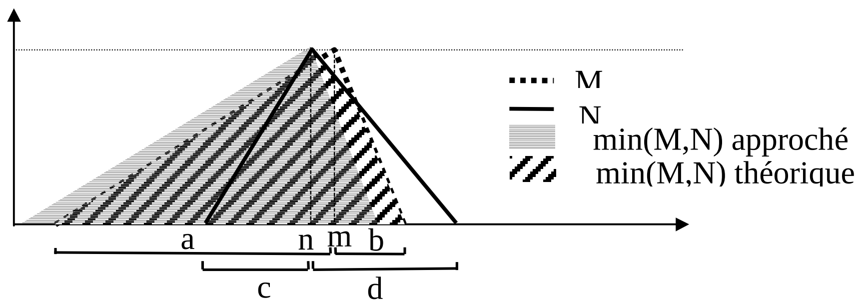


Fig. 36 : opérateurs flous de minimum théorique et approché

III.1.5 Recherche d'une solution d'un FCSP

L'objectif de la recherche d'une solution est de choisir des valeurs pour les variables de décision qui conduisent à la ou une des meilleures solutions, c'est-à-dire une instanciation d qui satisfait le problème à un degré égal à la consistance du problème (cons(P)=sat(d)). On va donc construire un arbre de recherche où chaque nœud correspond à l'instanciation d'une nouvelle variable de décision [Fargier 94]. Cet arbre est développé avec un algorithme de type depth first branch and bound. On peut l'améliorer en réalisant un filtrage, en employant des heuristiques [Guesgen 95.1] et en optimisant les retours arrière.

Le calcul du degré de satisfaction se fait de façon incrémentale à chaque instanciation d'une nouvelle variable. On arrête le développement de toute branche dont la consistance partielle est inférieure au degré α de satisfaction maximum des solutions trouvées jusqu'à présent. Il ne sert à rien de développer une branche qui conduit de façon certaine à une solution moins bonne que celles trouvées précédemment.

III.1.6 Conclusion

Pour la technique des gabarits, nous cherchons à exprimer des contraintes de temps mal connues ou décrites de façon imprécise. Ce sont donc des contraintes souples qui nous intéressent. On peut éventuellement utiliser aussi des contraintes avec priorités si, dans sa description du gabarit, l'expert précise que certaines contraintes temporelles sont plus importantes à vérifier que d'autres. Les autres types de contraintes n'entrent pas dans le cadre de notre application. On exprimera ces contraintes par des nombres flous plutôt que des relations floues qui augmentent trop la complexité algorithmique.

L'extension des CSP aux FCSP telle qu'elle est présentée ici implique de fait une vision égalitariste pour l'évaluation. En effet les degrés de consistance d'un problème et d'une solution sont calculés en réalisant l'intersection des contraintes, ce qui correspond bien à la contrainte la moins satisfaite. Cette évaluation n'est pas assez discriminante dans certains cas, mais nous n'utiliserons pas la satisfaction globale du problème. Nous voulons seulement utiliser les fenêtres temporelles mises à jour pour calculer une compatibilité partielle entre un élément du scénario et les observations associées. C'est l'agrégation de ces compatibilités partielles fixée par la structure du gabarit qui permettra de raffiner l'évaluation.

Par rapport au problème de reconnaissance de scenarii, le concept de graphe minimal est apparu comme essentiel pour l'actualisation des fenêtres et des contraintes temporelles. Mais la littérature n'offre pas d'application réelle du calcul de ce graphe minimal dans le cas flou. Pour une application du type ordonnancement comme celle que traite Fargier on cherche une solution par des algorithmes du type branch and bound avec éventuellement un filtrage par 2-consistance et des heuristiques.

On dispose cependant des fondements théoriques pour le calcul du graphe minimal avec l'extension des algorithmes de type PC1 ou Ford. Mais il reste à les tester effectivement et à choisir la modélisation floue la plus adaptée. Faut-il faire des approximations dans les opérations élémentaires ? Que choisir entre les algorithmes de type Waltz et de type Floyd-Warshall ? Nous allons répondre à ces questions dans la suite tout en appliquant ces techniques à la reconnaissance de gabarits.

III.2 Représentation d'un gabarit par un graphe temporel flou

Dans un premier temps, il faut déjà établir une traduction entre la structure du gabarit représenté sous la forme d'une décomposition hiérarchique et une représentation temporelle du gabarit par un graphe de contraintes. Nous allons voir comment s'établit cette traduction sur un exemple simple.

III.2.1 Gabarit temporel classique

Actuellement, le formalisme des gabarits permet de considérer quatre principaux types d'éléments temporels, tous représentés sous une forme numérique. On peut exprimer la durée de chaque sous-activité, des délais absolus (Δ_{abs}), durées entre le début d'un événement et le début d'un événement fils, des délais relatifs (Δ_{rel}), durées entre la fin d'un événement et le début d'un événement frère. De façon implicite, on sait que la fin (resp. le début) d'un élément père de gabarit doit correspondre avec la fin la plus tardive (resp. le début du premier) de tous ses événements fils. Ces notions sont représentées sous leur forme hiérarchique classique sur la figure ci-dessous avec une interprétation graphique possible des contraintes temporelles : P_2 débute P_1, F_0 termine P_1, F_1 débute P_2 et F_2 termine P_2.

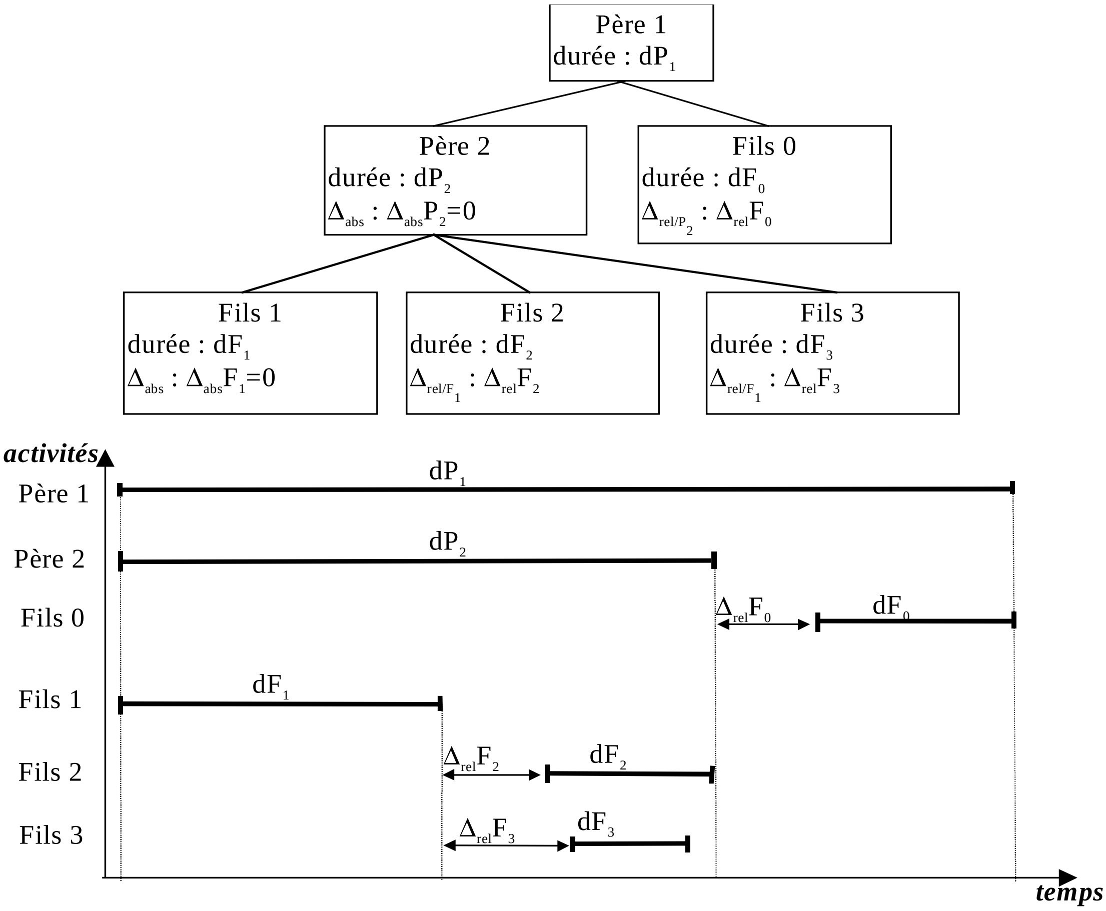


Fig. 37 : représentation hiérarchique et interprétation temporelle d'un gabarit

III.2.2 Graphe temporel correspondant

On peut aisément traduire dans un graphe temporel toutes les relations concernant des durées et des délais. Il suffit pour cela d’introduire pour chaque élément de gabarit sa date de début et sa date de fin ($X_{déb}$ et X_{fin} pour l'activité X). On introduit en plus une date de référence t_0 arbitrairement fixée à 0 qui permettra d'ajouter les informations sur les dates d'événements observés pour avoir le graphe du scénario actualisé.

Le graphe du scénario actualisé est en effet le graphe temporel correspondant au gabarit initial avec en plus des informations de dates, de durées ou de délais liées aux observations. Le graphe de la session a la même forme, il comporte les mêmes informations mais sans les contraintes du gabarit initial. On peut ainsi représenter tous les graphes temporels utiles.

On ne peut en revanche pas représenter explicitement le fait qu'un gabarit père doit toujours se terminer avec la fin de son fils le plus tardif, puisque l'on ne sait pas forcément à l'avance quel sera ce fils. Sur le schéma précédent par exemple, le père 2 peut se finir indifféremment en même temps que le fils 2 ou le fils 3. On verra par la suite comment on peut tenir compte de cette contrainte supplémentaire et de la contrainte réciproque sur le début des activités lors de l'obtention du graphe minimal. Chaque nœud du graphe doit en fait comporter des informations supplémentaires : la liste des nœuds correspondant aux dates de début et de fin de ses fils si c'est un nœud père et les nœuds correspondant aux dates de début et de fin de son père si c'est un nœud fils.

Voici une traduction sous forme de graphe du gabarit précédent. On ne représente que quelques arcs au départ de t_0 par souci de simplification, les autres arcs issus de t_0 peuvent se déduire par propagation.

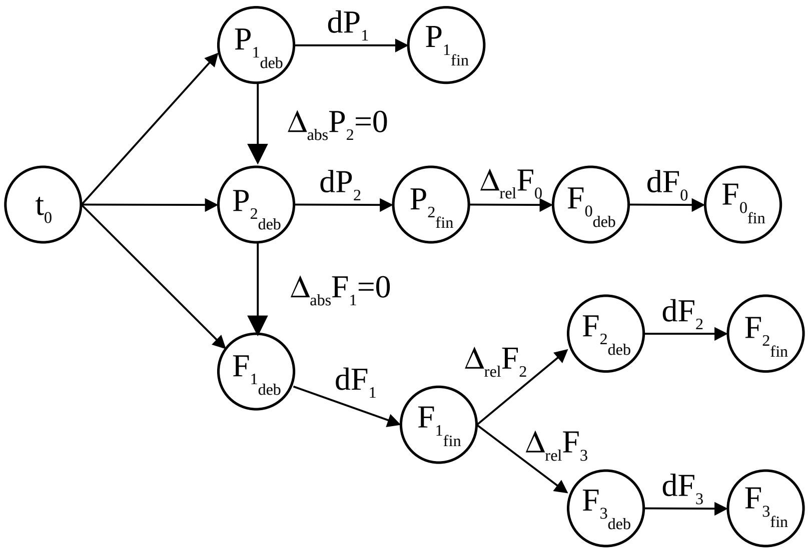


Fig. 38 : graphe temporel correspondant au gabarit initial

On peut même enrichir le formalisme en autorisant des délais entre le début ou la fin d'un élément et le début ou la fin de n'importe quel autre élément du même gabarit. Les arcs reliant t_0 aux autres nœuds du système sont en fait les inconnues du problème. Dès qu'ils sont connus ils fixent les dates des différents éléments.

III.2.3 Intégration d'observations

Au fur et à mesure que l'on instancie le gabarit par des observations, il faut d'une part enrichir le graphe temporel initial afin de construire le graphe du scénario actualisé et d'autre part établir le graphe de la session ne prenant en compte que les observations. Pour le graphe de la session, il suffit de traduire par un graphe les événements observés faisant intervenir soit une date, soit une durée, soit un délai. Pour le graphe du scénario actualisé, les observations intégrées doivent être cohérentes avec les contraintes déjà présentes et pour cela on réalise une intersection entre les anciennes contraintes et les contraintes issues de l'observation.

Voici un exemple d'événements que l'on pourrait intégrer au gabarit précédent :

- **Observation 1** : l'activité de type fils 1 de durée d_{O_1} se termine à la date t_{O_1}.
- **Observation 2** : elle est suivie d'une activité de type fils 3 après un délai Δ_{O_2}.
- **Observation 3** : une activité de type fils 0 débute avec un délai Δ_{O_3} après la fin de l'activité de type fils 1 de l'observation 1.

On remarquera que l'observation 3 donne une information sur le délai entre une activité de type fils 1 et une activité de type fils 3, ce qui ne correspond à aucune contrainte spécifiée dans le gabarit. Mais la représentation sous forme de graphe permet tout de même la prise en compte de cette information. Le graphe de la session correspondant est le suivant :

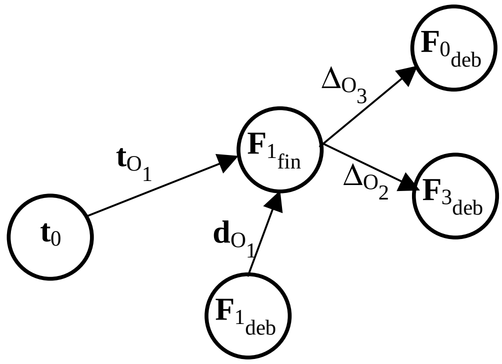


Fig. 39 : exemple de graphe de session

Enfin voici le graphe du scénario actualisé après intégration de ces informations. Les modifications par rapport au graphe initial apparaissent en grisé.

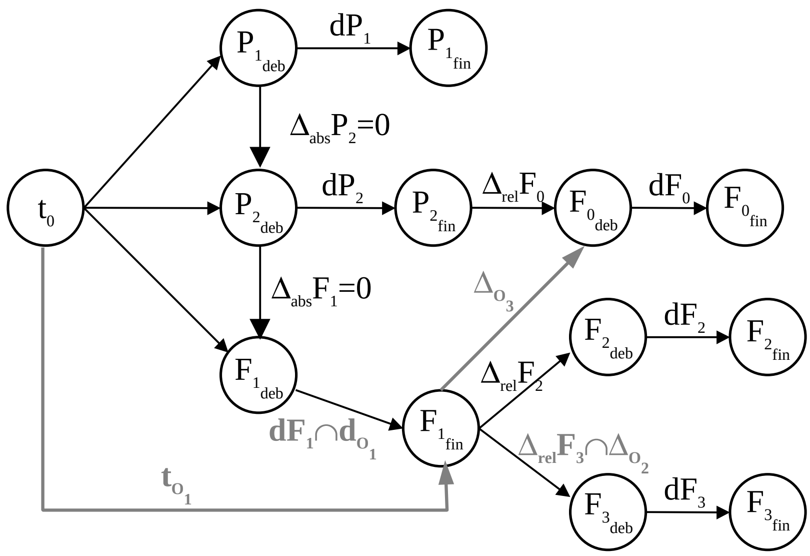


Fig. 40 : graphe du scénario actualisé

Jusqu'ici nous avons montré la représentation des contraintes temporelles pour les différents types de graphes, sans nous occuper de savoir si ces contraintes étaient cohérentes. Une fois ces graphes établis, il faut propager les contraintes afin de tenir compte de toutes les informations. La propagation optimale est obtenue avec le graphe minimal. Le calcul du graphe minimal est à peu près identique pour le scénario actualisé et pour la session, avec pour le scénario des contraintes supplémentaires liées à la décomposition hiérarchique.

III.3 Algorithme de reconnaissance

L'essentiel de ce chapitre est consacré au calcul du graphe minimal, fondement de notre algorithme de reconnaissance. Ce calcul dépend du choix que l'on retient pour la modélisation des ensembles flous, nous détaillons donc les intérêts et inconvénients des différentes modélisations, avec entre autres la possibilité ou non de traiter les contraintes disjonctives. L'algorithme est assez complexe mais on peut améliorer nettement ses performances par certaines heuristiques que nous présenterons. La partie évaluation de la reconnaissance n'est pas abordée ici et fera l'objet du chapitre IV p.131.

III.3.1 Rôle du graphe minimal dans la reconnaissance

Rappelons tout d’abord ce qu’est le graphe minimal. Un graphe temporel exprime des durées et des dates d’événements par des contraintes. La durée d’un événement est une contrainte entre le début et la fin de cet événement, et dater un événement correspond à donner une durée, donc une contrainte entre celui-ci et un événement fictif de date l’origine des temps. Chaque arc du graphe est étiqueté par une contrainte. Cette contrainte est vérifiée si les dates associées aux nœuds respectent la durée exprimée par cette contrainte.

Une solution globale du graphe est une instance de tous les nœuds qui vérifie toutes les contraintes. Une solution partielle est une instance de quelques nœuds seulement qui vérifie les contraintes entre les nœuds instanciés. Une solution partielle ne conduit pas forcément à une solution globale. Soit le graphe initial suivant, avec des contraintes sous la forme d'intervalles numériques pour plus de simplicité :

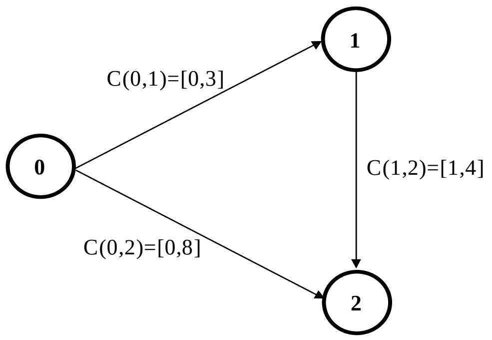


Fig. 41 : exemple de graphe

Si X_i représente la date du nœud i, une solution partielle est : $S_P=\{X_0=1;X_2=9\}$. $X_2-X_0=8$ vérifie bien C(0,2). Mais les deux autres contraintes C(0,1) et C(1,2) ne peuvent pas être vérifiées simultanément. En effet :

$$C(0,1) \Rightarrow 0 \leq X_1-X_0 \leq 3 \Rightarrow 1 \leq X_1 \leq 4$$

$$C(1,2) \Rightarrow 1 \leq X_2-X_1 \leq 4 \Rightarrow 5 \leq X_1 \leq 8$$

S_P ne peut pas être prolongée en une solution globale.

Le graphe minimal associé à un graphe G est un graphe équivalent à G (qui garde exactement les mêmes solutions globales) pour lequel toute solution partielle peut être prolongée en une solution globale. C'est un graphe dans lequel les contraintes sont les plus restrictives possibles, elles contiennent toutes et uniquement les solutions globales du graphe. Le graphe minimal correspondant à l'exemple précédent est :

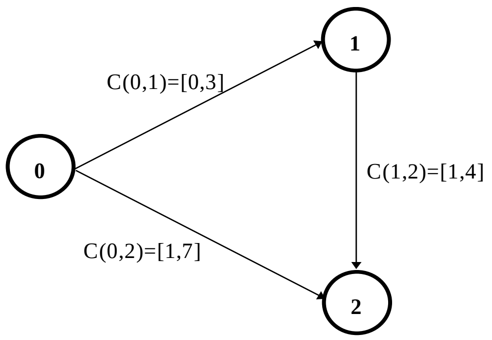


Fig. 42 : graphe minimal

S_P n'est plus une solution partielle puisque $X_2-X_0=8$ ne vérifie plus C(0,2). Grâce à ce graphe, on peut éliminer directement cette solution, ce qui était moins évident avec le graphe initial.

Si le graphe minimal comporte une seule contrainte vide, c'est qu'il n'y a aucune solution qui vérifie toutes les contraintes, le graphe est incohérent. Si au contraire toutes les contraintes sont non vides, on est sûr qu'il existe au moins une solution.

Cette notion de graphe minimal est primordiale pour la technique des gabarits. Un gabarit est un cadre général qui décrit un scénario typique. Sur le terrain d'activités, les informations sont partielles et incomplètes, et on cherche d'une part à savoir à partir de quelques données, si le scénario qui est en train de se dérouler correspond à un des scenarii typiques modélisés sous forme de gabarits, et d'autre part on voudrait prévoir et anticiper les événements à venir.

On peut comparer les données issues de l'observation avec les graphes minimaux correspondant aux différents gabarits. Si ces données sont une solution partielle d'un graphe, alors on peut conclure que la scène observée correspond peut-être au scénario modélisé par ce graphe, sinon le scénario est exclu. On peut aussi intégrer les données dans le graphe sous la forme de nouvelles contraintes et recalculer le graphe minimal du scénario actualisé, ce qui va donner toutes les dates possibles pour les événements encore non observés dans le cas où le scénario serait respecté jusqu'au bout. Le graphe minimal du scénario actualisé fournit toutes les informations indispensables, que ce soit pour l'analyse de la situation ou pour la prédiction.

Reprenons le graphe donné en exemple et supposons qu'il traduise un gabarit. On observe le fait X_0 au temps t_0=0 et le fait X_1 au temps t_1=2. En intégrant ces nouvelles informations dans le graphe, on obtient un nouveau graphe minimal du scénario actualisé :

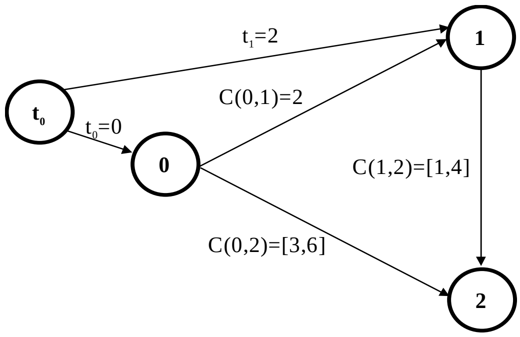


Fig. 43 : scénario actualisé

Le graphe minimal du scénario actualisé ne comporte pas de contrainte vide, les événements observés peuvent donc correspondre au scénario modélisé. On peut aussi déduire que $X_2 \in [3,6]$, ce qui est plus précis que l'information que l'on avait au départ et on peut donc se préparer à observer cet événement entre t=3 et t=6.

Il faut maintenir en parallèle le graphe minimal de la session qui permettra, par comparaison avec le graphe du scénario actualisé, de calculer les degrés de compatibilité partielle utilisés pour l'évaluation de la reconnaissance. Il est nécessaire de propager aussi les contraintes du graphe de la session afin d'assurer la cohérence et une précision maximale des informations. Si l'on a par exemple les observations : X_1 débute à t=5, X_1 dure entre 10 et 15 unités de temps et se termine entre t=20 et t=25 ; implicitement on sait que X_1 se termine à t=20 et dure exactement 15 unités de temps. C'est le graphe minimal de la session qui nous permet de tenir compte de ces précisions supplémentaires.

Finalement, voici l'algorithme global de reconnaissance :

Algorithme global de reconnaissance

- 0. compléter et calculer les graphes minimaux de tous les gabarits
- 1. pour chaque nouvelle contrainte C issue d'une observation associée à un gabarit de graphe G faire :
- 2. C correspond à une contrainte C_{anc} de G (graphe complet)
- 3. $C_{nouv} = C_{anc} \cap C$
- 4. si $C_{nouv} = \varnothing$ alors
- 5. l'observation ne peut pas instancier G, retour en 1
- 6. sinon l'observation instancie le gabarit, remplacer C_{anc} par C_{nouv} dans G
- 7. calcul du nouveau graphe minimal de G (ne peut avoir de contrainte vide, car on intègre une contrainte cohérente avec l'ancien graphe minimal)
- 8. ajout de C dans le graphe de la session
- 9. calcul du nouveau graphe minimal de la session G_{ses}
- 10. comparer G et G_{ses} pour évaluer la reconnaissance du gabarit considéré
- 11. retour en 1

Nous allons voir maintenant comment calculer en pratique le graphe minimal dans le cas de contraintes floues. Nous avons comparé différents algorithmes en utilisant plusieurs modélisations d'ensembles flous.

III.3.2 Modélisation des ensembles flous

A) Différentes représentations

On choisit souvent des fonctions de type trapézoïdal pour effectuer des calculs flous, car ce sont les plus simples à manipuler : seulement 2 intervalles sont nécessaires pour les définir. Mais même avec ces fonctions simples, on rencontre le problème suivant : le résultat d'une opération n'est pas forcément un ensemble trapézoïdal.

L’intersection ou le minimum flou de deux fonctions trapézoïdales normalisées ne donne pas forcément une fonction trapézoïdale normalisée, comme l’illustre le dessin ci-dessous.

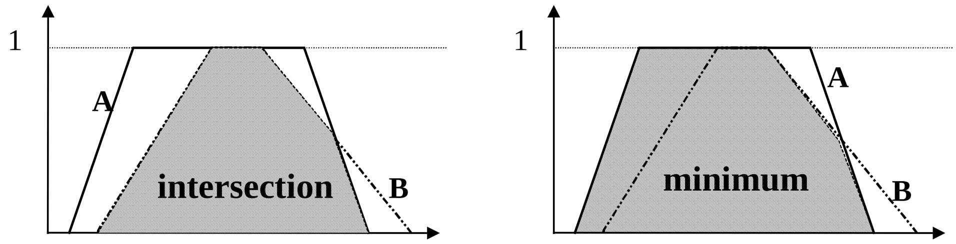


Fig. 44 : intersection et minimum de deux trapèzes

Que l’on utilise un algorithme de type consistance de chemins ou de type plus court chemin, il faudra résoudre le problème des opérateurs flous, soit en faisant des approximations, soit en augmentant la complexité et la précision de la modélisation.

Nous avons choisi d’utiliser 3 types de modélisations pour tester et comparer ces algorithmes. Leur choix est lié à la possibilité d’effectuer les opérations requises par les algorithmes, c’est-à-dire l’intersection, la somme, la différence et le minimum flous. Il y a aussi une progression dans la précision permise qui s’accompagne d’une complexité croissante. On ne considère pour l’instant que des ensembles flous convexes. Nous verrons le cas non convexe dans le cadre de l’étude de la disjonction.

La première modélisation est la représentation par des trapèzes. C’est la plus simple mais aussi la moins précise. Elle ne conserve que le support et le noyau de l’ensemble quelconque initial. On entend ici le terme noyau au sens $\{x \,/\, \mu(x)=h\}$ où h est la hauteur de l'ensemble puisque nous ne travaillons pas forcément avec des ensembles normalisés. M=(a,b,c,d,h) est un ensemble flou de noyau [a,b], de support [a-c,b+d] et de hauteur h. Voici l’approximation d’un ensemble flou quelconque par un trapèze :

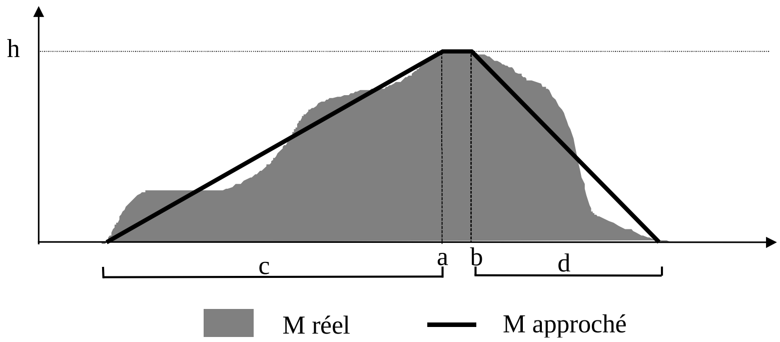


Fig. 45 : approximation d'un ensemble flou par un trapèze

Une deuxième représentation possible utilise les α-coupes. On peut moduler la précision liée à cette représentation en utilisant un nombre plus ou moins grand de α-coupes. Le plus souvent, on utilise 10 α-coupes échelonnées de 0,1 en 0,1. Le nombre flou M est donné par l'ensemble de ses n α-coupes : M={[a_i,b_i], i=1 à n}. Voici ce que cela donne sur l'exemple précédent :

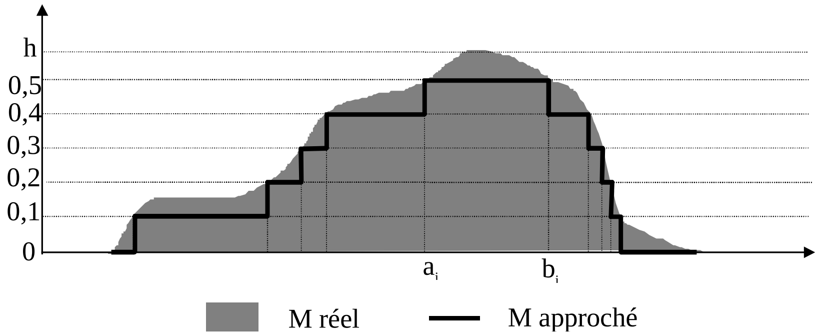


Fig. 46 : approximation d'un ensemble flou par 10 α-coupes

Enfin, la dernière représentation, la plus précise, consiste à approcher tout ensemble flou par des fonctions affines par morceaux continues. C'est la seule représentation qui permet d'avoir des ensembles non convexes, c'est-à-dire disjonctifs, ce qui peut être important comme nous le verrons par la suite. La précision de la représentation obtenue dépend du nombre de segments utilisés. Un nombre flou n est défini par une liste de points (x_i,y_i).

Une représentation possible de l'exemple précédent est :

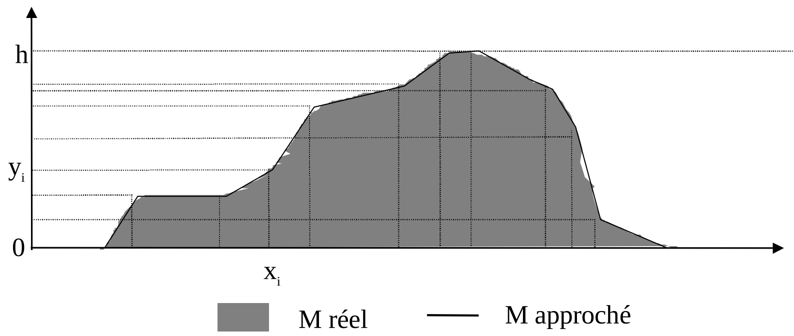


Fig. 47 : approximation d'un ensemble flou par des fonctions affines par morceaux

A noter que nous avons discrétisé l'axe des abscisses qui modélise le temps. C'est en effet beaucoup plus simple pour tous les aspects calculatoires tout en restant largement assez précis. L'axe est gradué toutes les minutes et la précision des données temporelles dans les gabarits ne dépasse jamais la minute.

B) Opérateurs

Pour ces 3 types de modélisation, nous avons redéfini les opérations nécessaires pour les algorithmes de propagation. Ces opérations sont la somme floue, l'intersection et le minimum flous. Toutes les autres opérations qui pourraient servir par la suite se déduisent de celles-ci. L'intersection est donnée par $\mu_{A\cap B}(x) = \min(\mu_A(x), \mu_B(x))$ si on choisit le minimum comme t-norme. Rappelons le principe d'extension qui est à la base de la définition de toutes les opérations sur des nombres flous. Soit un opérateur binaire f appliqué aux nombres flous A et B de fonctions d'appartenance μ_A et μ_B :

$$\mu_{f(A,B)}(x) = \sup_{\{y,z/x=f(y,z)\}} \left[\min(\mu_A(y), \mu_B(z))\right]$$

Mais lorsque l'on applique ce principe sur certaines catégories d'ensembles flous et avec certains opérateurs, le résultat n'est pas forcément de la même catégorie (cf. Fig. 44). Dans ce cas, on a essayé d'approcher le plus possible le résultat par un exemple de la même catégorie. De plus, si les ensembles ne sont pas normalisés, le degré d'appartenance maximum (hauteur) du résultat d'un opérateur est le minimum des hauteurs de chaque opérande.

Tout ce qui est au-dessus de ce seuil n'a aucune influence dans le calcul et par la suite on supposera qu'avant tout calcul les ensembles flous ont été aplatis à ce niveau. On peut ainsi travailler comme si les ensembles étaient normalisés. Sur la figure ci-dessous, B est aplati au niveau de A pour donner B'. On applique l'opérateur sur A et B'.

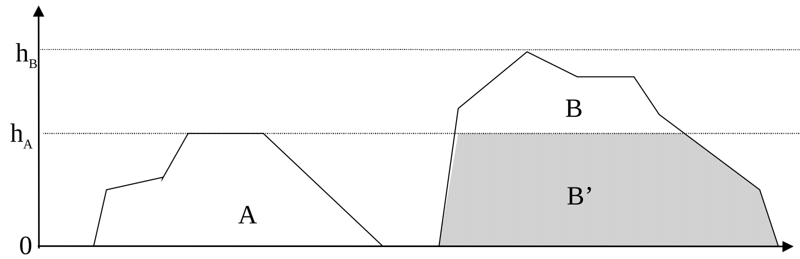


Fig. 48 : ensembles aplatis au même niveau

a) Trapèzes

i) Intersection

L'intersection de deux trapèzes ne donne pas forcément un trapèze si on applique :

$$\mu_{A\cap B}(x) = \min(\mu_A(x), \mu_B(x))$$

Nous avons choisi de construire le trapèze à partir du noyau et du support du résultat. Il semble en effet raisonnable d'assurer que les valeurs qui satisfont totalement ou pas du tout une contrainte soient préservées.

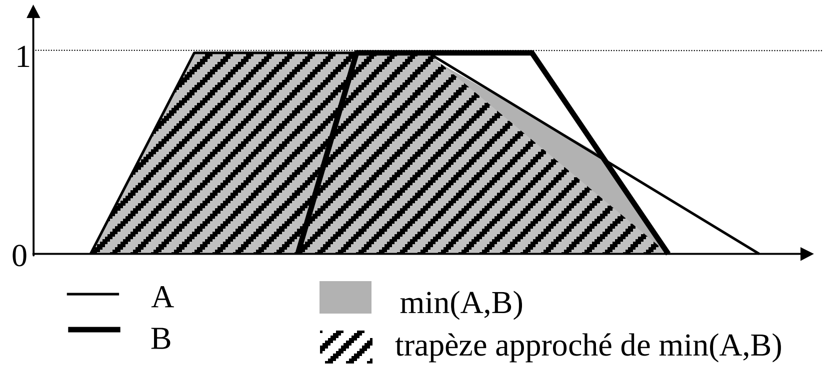


Fig. 49 : intersection de trapèzes

ii) Minimum

On applique exactement le même raisonnement pour l'opérateur minimum. On ne peut pas déduire du principe d'extension une formule simple pour le minimum. Après approximation, on obtient la formule suivante qui avait été donnée par Dubois et Prade [Dubois 93] et qui conserve le noyau et le support du résultat (contrairement à la formule donnée en III.1.4B) :

A=(a,b,c,d) B=(a',b',c',d')

min(A,B)=(min(a,a'),min(b,b'),min(a,a')-min(a-c,a'-c'),min(b+d,b'+d')-min(b,b'))

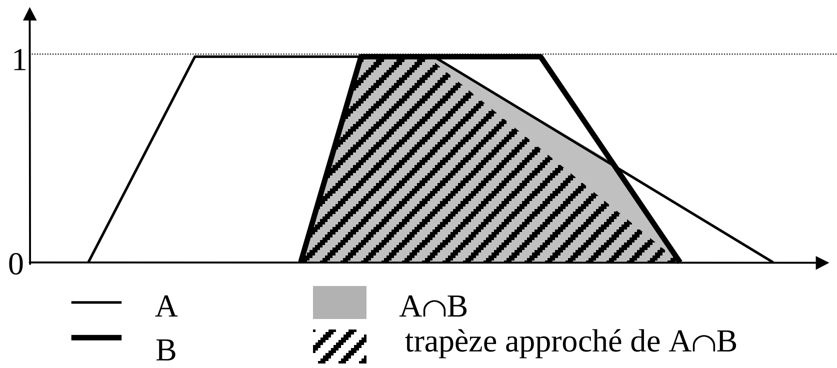


Fig. 50 : minimum de trapèzes

iii) Somme

D'après le principe d'extension, la somme floue de deux trapèzes est un trapèze et donc pour cet opérateur il n'est pas nécessaire d'introduire d'approximation. On peut simplifier la formule qui devient pour des trapèzes A=(a,b,c,d) et B=(a',b',c',d') :

A⊕B=(a+a',b+b',c+c',d+d')

b) α-coupes

Soit un nombre flou A de fonction d'appartenance μ_A. Son α-coupe de niveau α est définie par : $A_\alpha=\{x \,/\, \mu_A(x)\geq\alpha\}$. C'est un intervalle $[a_{1_\alpha}, a_{2_\alpha}]$ pour des ensembles convexes, ce qui est notre cas. Cette modélisation par α-coupes permet en fait de se ramener à des opérations sur les intervalles des α-coupes, ce qui est souvent beaucoup plus simple. Ceci d'après le théorème décrit dans [Dubois 80] :

$$[f(A,B)]_\alpha=f(A_\alpha,B_\alpha) \text{ ssi } \sup_{\{y,z/x=f(y,z)\}}\left[\min(\mu_A(y),\mu_B(z)\right] \text{ est atteint pour tout x.}$$

Cette condition est vraie pour les opérateurs que nous considérons si les α-coupes sont des intervalles fermés [Dubois 89]. Ceci n'est pas toujours vrai en théorie puisque pour modéliser des notions comme "après" on utilise la borne infinie. Dans la pratique, l'univers du temps est borné par des nombres raisonnablement grands et tout se passe comme si l'univers était fini. Le théorème s'applique donc et on a avec $A_\alpha=[a_{1_\alpha}, a_{2_\alpha}]$ et $B_\alpha=[b_{1_\alpha}, b_{2_\alpha}]$:

$$(A\oplus B)_\alpha=[a_{1_\alpha}+b_{1_\alpha},a_{2_\alpha}+b_{2_\alpha}]$$

$$(\min(A,B))_\alpha=[\min(a_{1_\alpha},b_{1_\alpha}), \min(a_{2_\alpha},b_{2_\alpha})]$$

$$(A\cap B)_\alpha=[a_{1_\alpha},a_{2_\alpha}]\cap[b_{1_\alpha}, b_{2_\alpha}]=[\max(a_{1_\alpha},b_{1_\alpha}), \min(a_{2_\alpha},b_{2_\alpha})]$$

c) Fonctions affines par morceaux

i) Intersection

L'intersection est toujours donnée par la formule $\mu_{A\cap B}(x)=\min(\mu_A(x),\mu_B(x))$. Pour l'appliquer, nous procédons en plusieurs étapes. Nous illustrerons ces étapes à l'aide des ensembles flous A et B suivants :

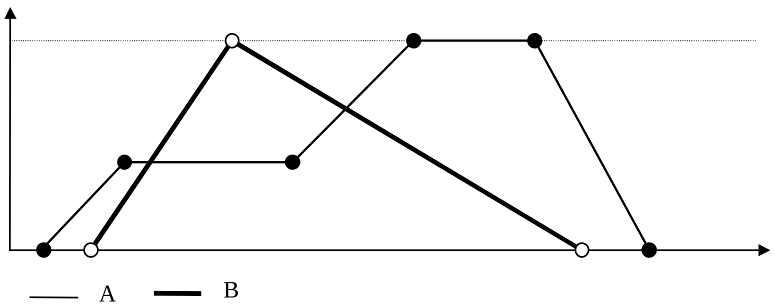


Fig. 51 : ensembles initiaux

- étape 1 : on discrétise A et B sur l'union de leur support, c'est-à-dire que pour tout couple (x,y) décrivant A on calcule l'ordonnée de B en cette même abscisse x et de même pour A avec les abscisses de B. On parvient à la description suivante : $A=\{(x_i,a_i)$, i=1 à n$\}$ et $B=\{(x_i,b_i)$ i=1 à n$\}$.

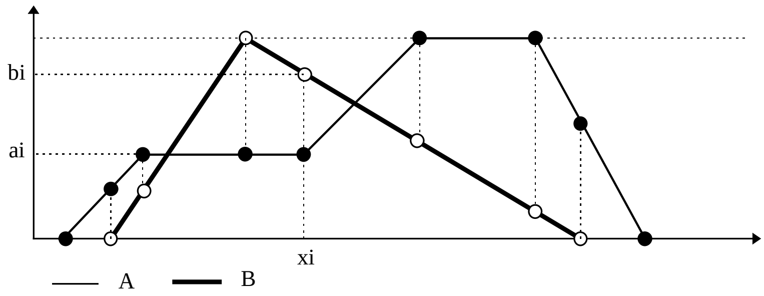


Fig. 52 : intersection - étape 1

- étape 2 : on calcule l'ensemble des points d'intersection de A et B. $I=\{(u_j,v_j)$, j=1 à m$\}$.

- étape 3 : un premier résultat est donné par $C=\{(x_i,\min(a_i,b_i))$, i=1 à n$\}\cup I$.

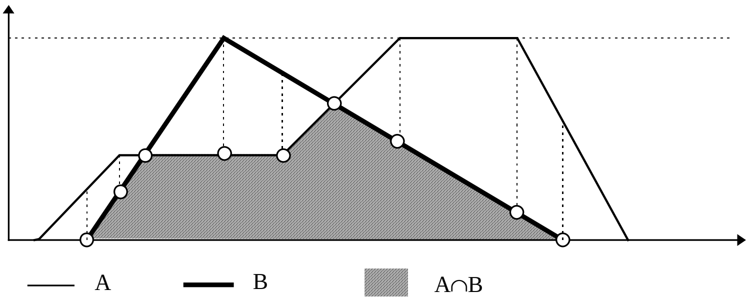


Fig. 53 : intersection - étape 3

- étape 4 : on simplifie le résultat en supprimant tous les points redondants. Un point est redondant s'il est aligné avec deux autres points, l'un d'abscisse inférieure et l'autre d'abscisse supérieure.

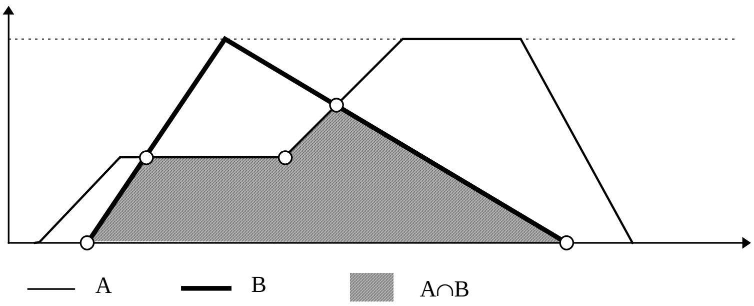


Fig. 54 : intersection - étape 4

ii) Somme et minimum

Dubois [Dubois 80] donne un algorithme généralisé de calcul des opérations binaires sur des nombres flous. Cet algorithme est déduit du principe d'extension et il s'applique pour tout opérateur croissant ou décroissant (la somme et le minimum sont croissants) avec des ensembles flous continus et pas forcément convexes. Le principe de l'algorithme est de décomposer chaque ensemble flou en ensembles flous élémentaires sur lesquels on sait appliquer l'opérateur et de combiner les résultats de ces opérations élémentaires. Voici le détail de cet algorithme appliqué à des fonctions affines par morceaux et pour l'opérateur minimum :

- étape 1 : on décompose chaque ensemble flou en deux ensembles : les segments non croissants et les segments non décroissants. Les segments constants font partie des deux ensembles.

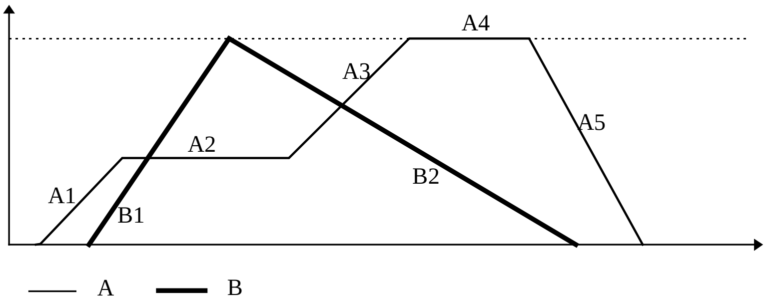


Fig. 55 : minimum - étape 1

segments non décroissants de A : A_{cr}={A1,A2,A3,A4}

segments non croissants de A : A_{dec}={A2,A4,A5}

segments non décroissants de B : B_{cr}={B1}

segments non croissants de B : B_{dec}={B2}

- étape 2 : on calcule le minimum des segments deux à deux de A_{cr} et B_{cr} et de A_{dec} et B_{dec} d'après le principe d'extension. min(A_i,B_j)=C_{ij}.

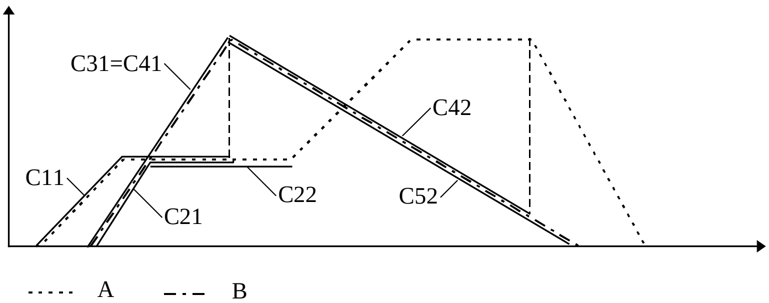


Fig. 56 : minimum - étape 2

- étape 3 : le résultat est obtenu en prenant le maximum (non flou) de tous les segments :

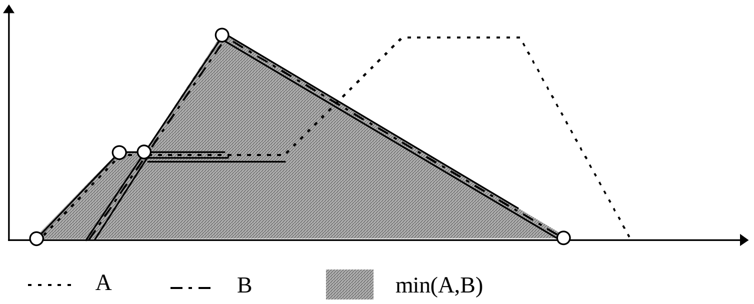


Fig. 57 : minimum - étape 3

On procède exactement de la même façon pour calculer la somme floue : simplement on remplace le min flou à l'étape 2 par une somme floue de chaque couple de segments. Cette somme floue s'obtient aussi relativement aisément en appliquant le principe d'extension.

III.3.3 Calcul du graphe minimal

A) Vérification de la cohérence

Comme on l'a vu précédemment, les algorithmes de calcul du graphe minimal sont fondés sur le filtrage par consistance des chemins de longueur 3. Par rapport aux algorithmes classiques présentés, il nous faut rajouter certaines contraintes bien spécifiques à l'application à la technique des gabarits.

Lorsque l'on décompose une activité en plusieurs sous-activités, il est en effet implicite que la première des sous-activités commence en même temps que son activité mère et que la dernière sous-activité se termine aussi en même temps que l'activité mère. Cette contrainte ne peut pas s'exprimer sous la forme d'une contrainte binaire du graphe.

Il faut donc ajouter aux algorithmes classiques une étape de vérification de la cohérence des activités mère/fille, et ceci par rapport d'une part au début des activités et d'autre part par rapport à leur fin. Les notions de "avant" (resp. "après") utilisées ici sont obtenues à partir de l'opérateur minimum (resp. maximum) et de l'ensemble flou unitaire U de degré d'appartenance 1 sur tout l'univers.

avant(A)=min(U,A)

apres(A)=max(U,A)

Soit une activité mère M et ses n sous-activités filles $F_{i,}$, X_{deb} et X_{fin} étant respectivement les dates de début et de fin de l'activité X, la vérification de la cohérence par rapport au début des activités se décompose ainsi :

- 1. Aucune activité F_i ne doit débuter avant l'activité M :

  ➪ de i=1 à n faire $F_{i_{deb}} = F_{i_{deb}} \cap apres(M_{deb})$

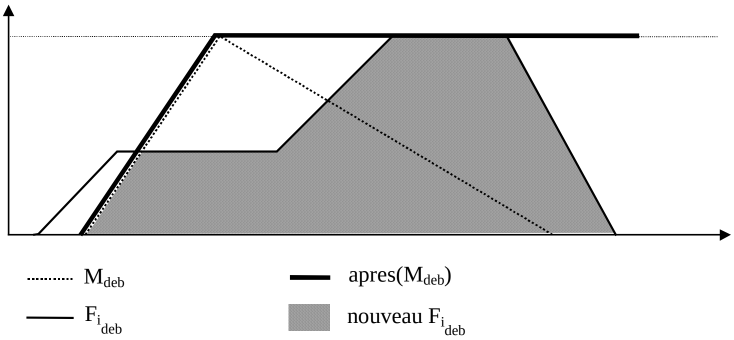


Fig. 58 : vérification de la cohérence - étape 1

- 2. L'activité M doit débuter avec le début au plus tôt des F_i.

  ➪ $M_{deb} = M_{deb} \cap (min_i(F_{i_{deb}}))$

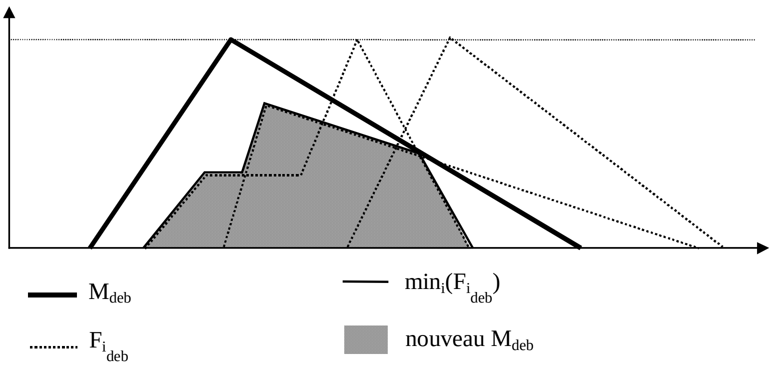


Fig. 59 : vérification de la cohérence - étape 2

On retrouve les mêmes étapes dans le même ordre pour la vérification de la cohérence de la fin des activités. Il suffit de remplacer les termes "après" par "avant" et le minimum par le maximum.

B) Différents algorithmes

Les différents algorithmes que nous avons utilisés résultent de combinaisons des algorithmes PC1, PC2, ou Floyd-Warshall avec ces étapes de vérification de cohérence. Nous allons énumérer ces différents algorithmes dont on comparera les résultats par la suite. Une seule version d'algorithme utilise le graphe de distances, la version Floyd-Warshall.

- Version **Floyd-Warshall et PC1** : les premiers algorithmes, les plus simples, consistent à appliquer soit PC1 soit Floyd-Warshall entièrement puis à vérifier ensuite sur tous les groupes d'activités mère/filles la cohérence de la fin et du début des activités. S'il y a des modifications lors de cette étape de vérification de la cohérence, on recommence l'algorithme PC1 ou Floyd-Warshall et ainsi de suite jusqu'à la stabilité.
- Version **PC2.0** : c'est exactement la même chose que précédemment en remplaçant l'algorithme PC1 par PC2. Tous les algorithmes suivants dérivent de PC2 qui est plus efficace que PC1.
- Version **PC2.1 avec p_1** : on organise l'ordre dans lequel les nœuds sont traités pour l'algorithme de filtrage PC2 suivant une priorité p_1. Les priorités sont établies en fonction de la topologie du problème. La priorité d'un nœud correspond à sa profondeur dans l'arbre, et les nœuds de plus grande priorité sont les nœuds les plus éloignés de la racine. La priorité du chemin (i,k,j) qui donne $C_{ij}=C_{ij}\cap(C_{ik}\oplus C_{kj})$ est la priorité du nœud j.
- Version **PC2.1 avec p_2** : la priorité p_2 est établie de façon un peu plus sophistiquée. On associe à chaque nœud la somme de son nombre total de descendants et de son nombre de frères. Les nœuds de numéro associé les plus faibles sont les plus prioritaires.

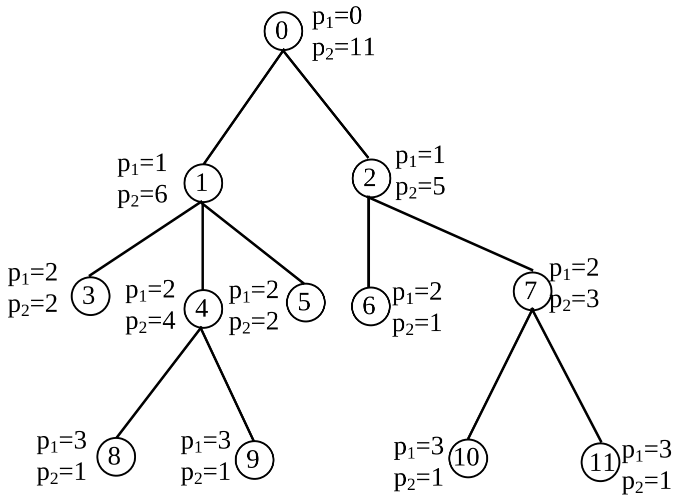


Fig. 60 : priorités p_1 et p_2

- **Version PC2.2** : ici, au lieu de vérifier systématiquement la cohérence pour tous les groupes mère/filles, on n'applique la vérification de cohérence qu'aux nœuds modifiés pendant le précédent déroulement de PC2. On tient à jour la liste des mères pour lesquelles il faut effectuer cette vérification. On applique le même degré de priorité pendant le déroulement de PC2 et pendant la vérification de cohérence, en prenant comme priorité pour un groupe mère/fille la priorité de la mère. PC2 est appliqué avec p_1 ou p_2.
- **Version PC2.3** : ici, on mélange complètement PC2 et la vérification de la cohérence. Dès qu'un nœud père ou fils est modifié, on effectue la vérification de cohérence qui correspond. PC2 peut être avec ou sans priorité, mais l'opération de vérification de cohérence est prioritaire sur tout.
- **Version PC2.4** : cette version mélange comme la précédente vérification de la cohérence et consistance de chemin. On tient à jour deux listes : celle des chemins pour lesquels il faut vérifier la consistance et celle des activités mères pour lesquelles il faut vérifier la cohérence par rapport aux filles. On traite en premier le nœud de plus grande priorité, et donc on alterne totalement les opérations de vérification de la consistance de chemin et de vérification de la cohérence en fonction de ces priorités.
- **Version PC2.5** : cette dernière version est très proche de la précédente. Au lieu d'avoir une seule liste, la liste des nœuds pères pour lesquels il faut effectuer une vérification de la cohérence complète, on maintient deux listes : celle des nœuds fils modifiés et celle des nœuds pères modifiés. On effectue une vérification partielle de la cohérence sur les nœuds des deux listes. Si c'est un nœud père qui a été modifié, on n'applique que l'étape 1 alors que si c'est un nœud fils, on applique l'étape 2 de la vérification de cohérence. On tient compte de la priorité des nœuds pour savoir s'il faut d'abord traiter un chemin, un nœud père ou un nœud fils.

Tous ces algorithmes travaillent sur un principe général commun, c'est seulement l'ordre des opérations et leur nombre qui varie selon des heuristiques. Pour la version Floyd-Warshall, l'opération élémentaire de propagation est aussi différente puisqu'elle effectue un minimum sur des distances au lieu d'une intersection de contraintes. Toutes les opérations de vérification de cohérence se font sur le graphe de contraintes. C_{0i} correspond à la date du nœud i, 0 étant le nœud arbitraire de début des temps.

On peut résumer ainsi ces algorithmes :

Algorithme général de calcul du graphe minimal :

- 0. Tant qu'une contrainte C_{ij} (resp. une distance d_{ij}) est modifiée et qu'aucune contrainte n'est vide faire :
- 1. Propager C_{ij} (resp. d_{ij}) pour tous les (k,j) : $C_{ik}=C_{ik}\cap(C_{ij}\oplus C_{jk})$ (resp. $d_{ik}=\min(d_{ik},d_{ij}\oplus d_{jk})$)
- 2. Si C_{ij} est la date de début ou de fin d'une activité mère $C_{0M_{deb}}$ ou $C_{0M_{fin}}$: appliquer l'étape 1 de vérification de cohérence
- 3. Si C_{ij} est la date de début ou de fin d'une activité fille $C_{0F_{deb}}$ ou $C_{0F_{fin}}$: appliquer l'étape 2 de vérification de cohérence

Par construction, on a bien un graphe 3-consistant qui vérifie la cohérence entre les débuts et les fins des activités et de leurs sous-activités. De plus, le graphe est complet (ou a été complété au départ) et la composition des contraintes est distributive sur l'intersection, donc d'après les théorèmes de Montanari on obtient bien le graphe minimal. Enfin, l'univers du temps est discrétisé et on considère qu'une contrainte floue est vide dès que sa hauteur passe en deçà d'un certain seuil, donc on est sûr que l'algorithme se termine en un temps fini. La complexité algorithmique dépend de la représentation choisie pour les ensembles flous.

Nous avons donc un certain nombre de résultats théoriques sur ces algorithmes. Nous avons voulu voir quelle était leur efficacité en pratique. Des séries de tests nous ont permis de faire des constats sur les différents choix de modélisation et sur l'apport des heuristiques.

III.4 Résultats

III.4.1 Comparaison des différentes modélisations

A) Comparaison qualitative

Nous avons effectué des comparaisons entre les différentes modélisations en simulant des petits scenarii sur des exemples simples de gabarits. Les contraintes initiales sont des trapèzes que l'on modélise selon les différentes catégories de représentation choisies. Au cours de la propagation, les ensembles prennent rapidement des formes plus complexes si la modélisation le permet.

Théoriquement, la modélisation par fonctions affines (liste de points) devrait donner des résultats exacts puisque les calculs respectent totalement le principe d'extension. Cependant on arrondit les résultats des calculs pour faciliter la représentation informatique et pour assurer que l'algorithme se termine : l'axe des abscisses est gradué en secondes donc on arrondit toutes les abscisses à la seconde près, et pour les ordonnées on peut régler la précision voulue. Ces arrondis introduisent des erreurs, mais cette représentation reste la plus précise et elle sert de référence pour les comparaisons.

Dans certains cas particuliers, on peut observer une très grande différence entre les résultats de la propagation pour des trapèzes et pour des listes de points. On peut mettre en évidence cette différence sur une seule étape de propagation. Supposons que l'on a les contraintes normalisées suivantes sous forme de trapèzes : C(0,1)=(91,110,1,20), C(0,2)=(150,170,80,80) et C(2,1)=(-50,-60,20,20). Alors C(0,2)⊕C(2,1)=(100,110,100,100).

Jusque là, les deux modélisations sont en accord, mais pour l'intersection C(0,1)∩[C(0,2)⊕C(2,1)], il faut faire une approximation sur les trapèzes et voici ce que donnent les deux résultats :

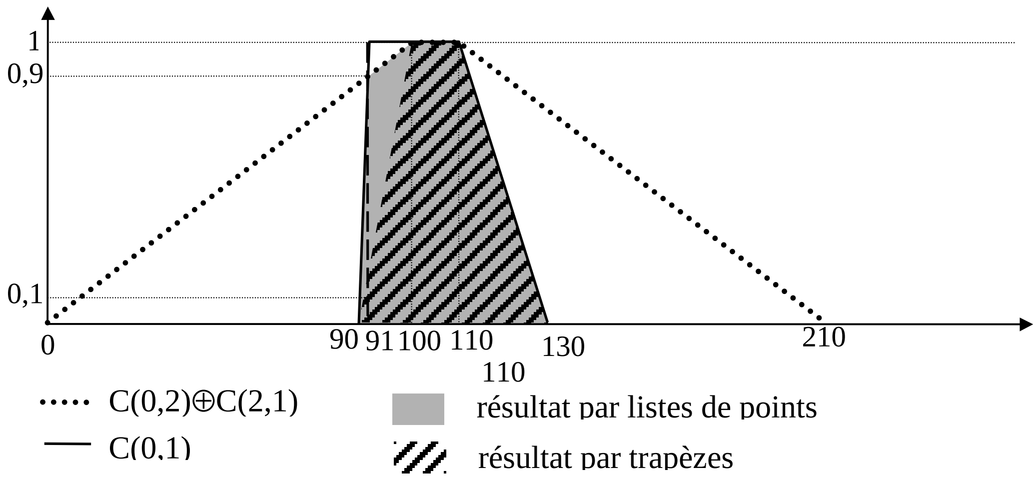


Fig. 61 : différence entre modélisation par trapèzes et par listes de points

La nouvelle contrainte C(0,1) obtenue par des trapèzes est très différente de celle obtenue par des listes de points. On a même une différence de 0,8 pour le degré d'appartenance en x=91.

Ce genre de situation se produit lorsque l'on combine deux trapèzes, l'un avec une pente très forte et l'autre avec une pente très faible. A priori, pour la définition des gabarits, chaque contrainte peut être définie avec plus ou moins de précision et on ne peut absolument pas prédire les degrés des pentes qui peuvent être hétéroclites. On risque alors de trouver ce type d'erreurs et il vaut donc mieux éviter la modélisation par trapèzes.

On a ensuite testé la modélisation par α-coupes. Les résultats sont très variables selon que l'on utilise une, cinq ou dix α-coupes. Si l'on souhaite une précision de 0,1, ce qui paraît raisonnable, il faut au moins 10 α-coupes, échelonnées régulièrement de 0,1 jusqu'à 1. Si l'on compare cette représentation avec celle par liste de points, les résultats sont dans l'ensemble très proches. On peut voir quelques exemples de résultats en annexe VII.3 p.248. Dans certains cas même, le passage d'une représentation à l'autre peut se faire simplement en prenant comme points les bornes des α-coupes.

On peut cependant trouver des décalages plus importants qui sont directement liés à la modélisation des contraintes initiales. Les α-coupes sont en effet données avec des bornes entières, ce qui fait perdre de la précision puisque certaines α-coupes vont être égales alors qu'elles ne le devraient pas.

Pour un ensemble triangle (rectangle particulier) de noyau 0 et de support [-1,1] par exemple, les α-coupes de niveaux 0,1 à 0,5 sont [-1,1] et les suivantes [0,0]. Ceci est illustré sur le schéma suivant :

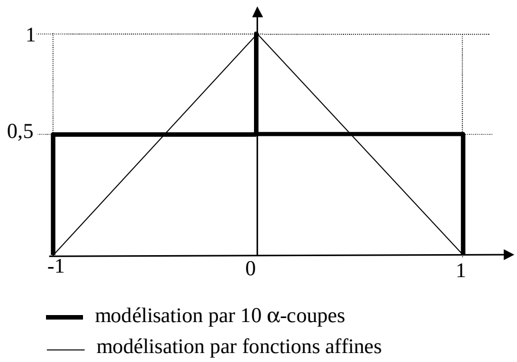


Fig. 62 : problème de la modélisation initiale par α-coupes

Avec la propagation, ce phénomène est amplifié et il arrive que les deux résultats obtenus pour des modélisations par liste de points et par α-coupes diffèrent de plus de 0,1 en certains points. Si, au départ, une contrainte n'a que 2 α-coupes différentes comme sur l'exemple précédent, on ne peut assurer qu'une précision de 0,5.

Un moyen efficace de remédier à ce problème est de modifier l'unité de discrétisation. Pour l'exemple précédent, si on passe à une discrétisation en dixième de secondes, le problème n'existe plus. Avec la modélisation par α-coupes, il faut donc bien faire attention aux ensembles de départ : si l'on est obligé de faire trop d'arrondis sur les α-coupes, c'est que l'échelle de discrétisation choisie est trop grande et il faut prendre une unité plus faible.

Nous avons vu que l'on pouvait régler la précision en ordonnée pour la modélisation par fonctions affines. Nous avons donc comparé les résultats obtenus pour des scenarii identiques en faisant varier la précision en y. Nous avons pris des précisions de 10^{-1} à 10^{-5} en diminuant d'un facteur 10 à chaque fois. L'algorithme de propagation s'arrête dès qu'il n'y a plus de modification à faire, c'est-à-dire dès que chaque nouvelle contrainte calculée par chemins-consistance est égale à la précédente. Cette notion d'égalité dépend de la précision choisie, et pour cette raison, plus la précision est faible, plus l'algorithme s'arrête vite.

Les résultats obtenus diffèrent donc légèrement. Sur toutes les simulations effectuées, on pouvait constater quelques différences entre une précision de 10^{-1} et 10^{-2} mais il n'y avait plus de différence à partir de 10^{-2} (cf. annexe VII.3.2 p.250). De toute façon, une précision de 0,01 est largement suffisante, surtout compte tenu du fait que les ensembles flous initiaux sont définis par expertise. Si l'on souhaite une précision finale de seulement 0,1, on peut se contenter d'une précision de 10^{-1} en y. Ce choix dépend des traitements qui sont ensuite effectués sur les degrés d'appartenance, mais la plupart du temps une précision de 0,1 conviendra.

Il peut arriver aussi pour certains types de gabarits que l'on ait des informations très précises sur certaines contraintes et floues sur d'autres. Il faut pouvoir mélanger des contraintes qui seront des singletons ou des intervalles classiques avec des ensembles flous traditionnels. Ceci ne pose pas de problème pour les modélisations par trapèzes et par α-coupes. Pour les fonctions affines en revanche, on ne pourra pas reproduire exactement un intervalle classique du fait que les discontinuités ne sont pas autorisées.

Un intervalle classique [a,b] peut se représenter par un trapèze (a,b,0,0) ou par des α-coupes valant toutes [a,b]. La liste de points la plus adéquate pour représenter cet intervalle est {(a-1,0),(a,1),(b,1),(b+1,0)}. Ce léger décalage peut se propager avec l'algorithme (cf. annexe VII.3.3 p.251). Mais en général, ce décalage ne jouera que sur quelques secondes. Les faits observés ne peuvent pas être datés à la seconde près la plupart du temps, et lors du calcul de la compatibilité des faits avec le gabarit, ce léger décalage devrait être imperceptible.

B) Comparaison des temps de calcul

Jusqu'ici nous avons analysé uniquement la qualité des résultats obtenus. Un critère déterminant est aussi le temps de calcul. La propagation doit en effet servir dans un système d'aide à la décision qui fonctionne en ligne, le traitement des informations ne doit donc pas prendre trop de retard par rapport à l'arrivée des données. Un fait observé pourra déclencher plusieurs hypothèses de gabarits donc plusieurs algorithmes de propagation en même temps. Il est important de veiller à limiter les temps de calcul. Nous avons donc comparé les temps de propagation pour des situations identiques représentées par des trapèzes, par une, cinq ou dix α-coupes ou enfin par des fonctions affines. L'algorithme utilisé est PC1, il sert de base de comparaison pour toutes les modélisations.

Voici un exemple caractéristique des résultats obtenus :

modélisation	trapèzes	1 α-coupe	5 α-coupes	10 α-coupes	fonctions affines
emps de calcul (s)	0,5	0,8	2	3	6

Un premier constat s’impose : plus on a une représentation précise, plus les temps de calcul sont longs, ce qui paraît logique. Il existe une seule exception à cette règle. Il apparaît sur la totalité des simulations effectuées que les temps de calcul sont légèrement plus longs pour une α-coupe que pour des trapèzes (même si ces temps de calcul sont très faibles tous les deux), alors que l'on calcule 4 points pour un trapèze au lieu de 2 pour une α-coupe, et théoriquement les temps de calcul devraient être inversés.

Cette exception vient certainement de la façon de stocker les données pour les modélisations : pour les α-coupes on utilise des tableaux dont la taille dépend du nombre de α-coupes, alors que pour les trapèzes on n’a que 4 entiers (a,b,c,d) et un réel pour la hauteur. La modélisation par trapèze est plus efficace parce que plus spécifique alors que la modélisation par α-coupe est plus générique puisqu’elle est valable quel que soit le nombre et le niveau des α-coupes. La modélisation par une α-coupe est moins précise que celle par des trapèzes qui paraît déjà insuffisante, donc elle ne présente pas d'intérêt.

Fargier [Fargier 94] a supposé que les temps de calcul pour des représentations par α-coupes seraient proportionnels au nombre de ces α-coupes. Ceci n’apparaît pas sur les résultats présentés. En effet, si on a beaucoup de α-coupes pour un ensemble flou, elles ne sont pas forcément toutes significatives et elles ne seront pas obligatoirement toutes modifiées. Dans le pire des cas, si toutes les α-coupes sont modifiées en même temps, on aura un temps de calcul pour 10 α-coupes 2 fois plus grand que pour 5 α-coupes, mais ceci arrive rarement.

Enfin, pour les fonctions affines, les temps de propagation sont toujours plus importants que pour 10 α-coupes même si parfois il y a moins de points à calculer pour les fonctions affines par morceaux simples. Mais le traitement des fonctions affines est plus complexe puisqu’il faut pour chaque opérateur somme ou minimum les décomposer en segments et combiner les segments deux à deux puis prendre leur maximum.

Nous allons maintenant regarder de plus près les modélisations par 10 α-coupes et par listes de points avec différents réglages de précisions. Ce sont en effet les seules représentations qui permettent d’atteindre la précision raisonnable de 0,1. L’algorithme utilisé pour ces tests n’est pas optimal, il sert simplement à comparer les différentes modélisations. Une première comparaison est effectuée pour le calcul d’un graphe minimal pour un gabarit relativement petit après introduction d’une information nouvelle. Le calcul du graphe minimal inclut aussi les étapes de vérification de la cohérence entre les nœuds père et fils.

	10 α-coupes	fonctions affines			
précision	/	0,1	0,01	0,001	0,0001
temps de calcul (s)	4	11	26	55	60

On voit que les temps de calcul augmentent de façon très importante, alors que comme on l'a vu à partir d'une précision de 0,01 les différences entre les résultats sont tout à fait négligeables. On supprime donc les précisions supérieures à 0,01. Il faut encore choisir entre les 3 modélisations restantes.

Nous avons construit pour poursuivre les tests un scénario complet sur un gabarit de très grande taille (cf. annexe VII.4 p.253). Le gabarit choisi comporte une quarantaine de nœuds, alors qu'en moyenne un gabarit est décrit par une dizaine d'activités, ce qui fait une vingtaine de nœuds pour son graphe de contraintes. Les temps de calcul obtenus sont donc plus importants que ce que l'on aurait sur des gabarits réels, mais ils accentuent les diffférences entre les modélisations et facilitent les comparaisons.

A chaque étape du scénario on fournit deux nouvelles contraintes, ce qui correspond à l'observation d'un fait. Si on observe une activité reconnaissance météo à un temps t_0, on en déduit deux contraintes : le début de l'activité reconnaissance météo se situe au plus tard à t_0 et sa fin au plus tôt à t_0.

	10 α-coupes	fonctions affines	
		précision=0,1	précision=0,01
étape 1	18s	38s	38s
étape 2	13s	50s	41s
étape 3	27s	28s	34s
étape 4	12s	15s	15s
étape 5	10s	17s	24s
étape 6	7s	17s	25s
étape 7	2s	4s	4s
étape 8	2s	4s	7s
étape 9	4s	9s	13s
étape 10	1s	15s	15s

Quelle que soit la modélisation adoptée, les temps de calcul sont très variables tout au long du scénario. Il peut y avoir un facteur nettement supérieur à 10 entre l'étape la plus courte et la plus longue. Ceci s'explique par deux phénomènes. D'une part, la propagation d'une nouvelle information peut avoir très peu ou beaucoup de conséquences. Certaines contraintes ne vont entraîner aucune modification sur leurs voisines et la propagation atteint un état stable après seulement quelques boucles de propagation, alors que pour d'autres les modifications vont s'enchaîner et la totalité des contraintes pourra être modifiée.

D'autre part, la tendance générale est à une réduction des temps de calcul au fur et à mesure que l'on avance dans le scénario. En fait, dans le graphe initial, seules les contraintes relatives entre nœuds sont instanciées. Aucun événement ne peut être daté, on connaît seulement leur durée et les délais entre eux. Toutes les contraintes par rapport au temps t=0 choisi arbitrairement sont inconnues, donc elles sont représentées par l'ensemble flou unitaire de fonction d'appartenance 1 sur tout l'univers.

La première information se traduit par une contrainte par rapport à t=0 et donc par propagation on va, grâce à cette donnée, instancier un certain nombre de contraintes relatives à t=0. Il y aura donc beaucoup de contraintes à modifier dans les premières étapes de propagation, ce qui explique des temps de calcul plus longs. Plus on avance dans le temps, plus les contraintes deviennent précises, donc moins elles sont modifiées par une nouvelle information et plus l'algorithme est court.

Si l'on compare maintenant les différentes représentations entre elles, on constate que la modélisation par 10 α-coupes est réellement la plus efficace dans toutes les situations. Entre les fonctions affines de précision 0,1 et de précision 0,01, la différence est moins nette. C'est quasiment toujours avec une précision de 0,1 que l'algorithme est le plus rapide. L'algorithme devient stable à 0,1 près alors qu'il ne l'est pas encore à 0,01 près.

Dans certains cas on obtient les mêmes temps de calcul, ce qui correspond aux cas où les algorithmes s'arrêtent au même niveau et donnent exactement les mêmes résultats pour les deux méthodes. Enfin, dans un cas seulement, l'algorithme est plus rapide avec la précision de 0,01. Ceci est certainement lié au fait que l'algorithme n'est pas optimal, et en faisant une modification à 0,01 près sur une contrainte, modification qui n'est pas faite dans le cas de la précision 0,1, cela modifie complètement l'ordre des opérations. Or les temps de calcul dépendent très fortement de l'ordre des opérations comme on le verra par la suite. Dans ce cas précis, le changement de l'ordre des opérations est favorable malgré une complexité de calcul plus grande.

III.4.2 Comparaison des différents algorithmes

A) Graphe de contraintes ou de distances

Les premiers algorithmes que nous avons voulu comparer sont l'algorithme PC1 appliqué à un graphe de contraintes et l'algorithme de Floyd-Warshall appliqué à un graphe de distances. Les résultats des algorithmes sont parfaitement équivalents pour des ensembles flous représentés par leurs α-coupes. Pour des fonctions affines, les résultats diffèrent légèrement (cf. annexe VII.3.4 p.252). Les opérations étant différentes (minimum au lieu d'intersection), les calculs ne sont pas toujours arrondis de la même façon.

De plus, les gabarits sont stockés sous la forme de graphes de contraintes et non de distances, ce qui évite d'avoir deux étiquettes au lieu d'une pour chaque arc. Mais à chaque fois que l'on effectue un calcul sur les distances, il faut convertir la contrainte en deux distances et une fois le calcul terminé on fait l'intersection des nouvelles distances obtenues pour retrouver une contrainte. Cette opération d'intersection donne lieu à de nouveaux arrondis, et même à de nouvelles approximations si l'on utilise une représentation par trapèzes.

Les différences entre graphe de contraintes et de distances restent de l'ordre de la précision en y lorsque l'on a une modélisation par fonctions affines, mais elle est plus importante pour des trapèzes. Voici ce que l'on peut obtenir en une étape de l'algorithme :

Au départ, nous avons les contraintes suivantes :

C(0,3)=(130,130,30,10,1) C(3,6)=(80,100,20,20,1) C(0,6)=(190,200,10,10,1)

Si l'on applique l'algorithme PC1, on obtient C(0,6)=C(0,6)$\cap$(C(0,3)$\oplus$C(3,6)) :

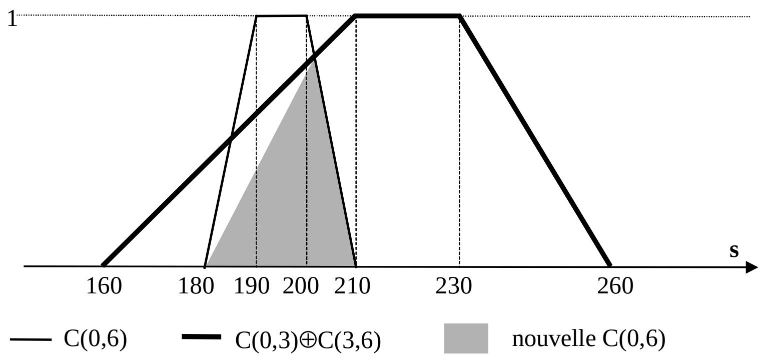


Fig. 63 : calcul par graphe de contraintes

Alors que si l'on applique Floyd-Warshall : d(0,6)=min[d(0,6),(d(0,3)⊕d(3,6))] et d(6,0)=min[d(6,0),(d(3,0)⊕d(6,3))] avec C(0,6)=d(0,6)∩[-d(6,0)].

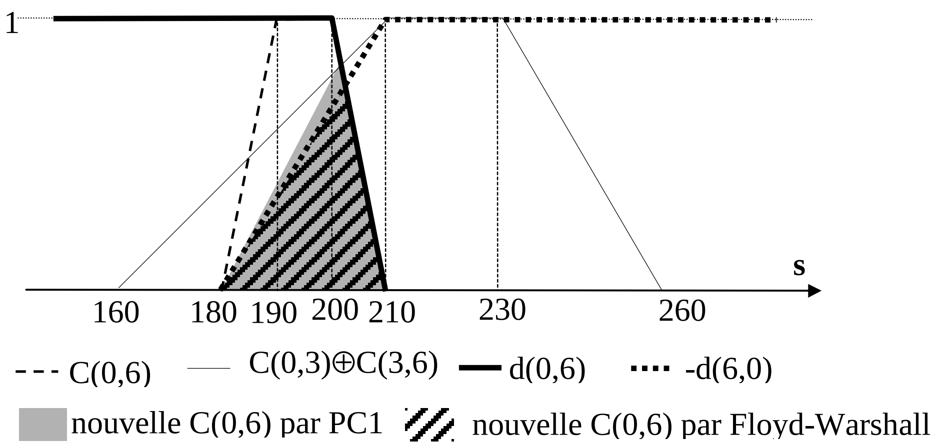


Fig. 64 : calcul par graphe de distances

Dans cet exemple, on arrive à une différence de 0,1 en x=202s entre les deux méthodes. Puis cette différence peut se propager et s'accentuer.

Du point de vue des temps de calcul, Floyd-Warshall est aussi plus long que PC1 quelle que soit la modélisation. Ceci est toujours lié au même problème : comme les gabarits sont sous forme de graphe de contraintes, les passages de contraintes à distances et inversement prennent du temps. Pour une modélisation par liste de points par exemple, on passe de 6 à 20 secondes, ce qui est considérable.

On pourrait tester l'algorithme de Floyd-Warshall en conservant les gabarits sous forme de graphes de distances, mais ceci créerait des structures plus complexes et plus coûteuses en mémoire et ne devrait pas apporter plus de précision pour les calculs puisque les approximations faites sont du même type pour les opérateurs somme et minimum. Par la suite, on abandonne les graphes de distances et on n'utilise plus que des algorithmes fondés sur PC1.

B) Temps de calcul

On cherche ici à optimiser l'algorithme de propagation. Les comparaisons sont toujours effectuées sur une même modélisation, la modélisation par fonctions affines. La comparaison des algrithmes est de toutes façons indépendante de la représentation choisie, et la modélisation par α-coupes étant la plus complexe elle permet de faire ressortir les différences. Nous avons testé les algorithmes PC1 et PC2 dans leur version la plus simple avec vérification de la cohérence sur un scénario quelconque. Il s'avère que le passage à PC2, qui permet de ne regarder que les chemins qui ont déjà été modifiés est très bénéfique. La propagation lors de la première étape passe de plus de 6 minutes à une trentaine de secondes et les suivantes de plus de deux minutes à une dizaine de secondes. La perte de temps liée à la gestion des chemins à vérifier est négligeable par rapport au gain en nombre d'opérations effectuées.

Nous avons ensuite repris pour optimiser PC2 l'exemple de gabarit et le scénario déjà utilisés pour les tests de comparaison du chapitre III.4.1B) p.105 et qui sont présentés en annexe VII.4 p.253. Nous avons essayé d'améliorer les performances en modifiant l'ordre des opérations, soit en ajoutant une notion de priorité, soit en intégrant la vérification de la cohérence dans la propagation à proprement dite. Le tableau donné en annexe VII.5 p.255 résume les résultats obtenus (les variantes des algorithmes ont été décrites en III.3.3B) p.99, en précisant pour chaque version de l'algorithme si l'on prend en compte un niveau de priorité p et si oui lequel.

Nous avons noté pour chaque version le temps de calcul par étape, puis la somme des temps de calcul pour ces 10 étapes (cf. Fig. 65). Nous avons aussi précisé le temps de calcul pour propager les 10 informations en une seule étape (notée "1 à 10") (cf. Fig. 66). Dans la réalité en effet, le système ne recevra pas forcément les informations une par une. Elles pourront être retardées et arriver en même temps que d'autres informations sur des événements différents. Rappelons que les données ne parviennent pas toujours par ordre chronologique au système.

Nous avons d’abord effectué une comparaison en ne considérant que la somme des temps de calcul des étapes 1 à 10 (cf. Fig. 65). C’est en effet le critère synthétique le plus déterminant. Il correspond au cas le plus fréquent où l’on reçoit les informations en petit nombre.

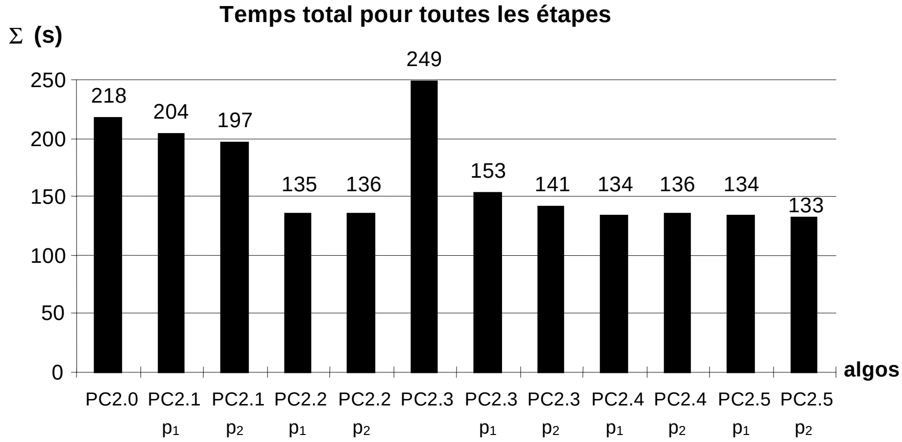


Fig. 65 : comparaison des temps de calcul des différentes versions de PC2

Un premier constat est l’efficacité de l’introduction d’une notion de priorité. Les deux algorithmes les plus mauvais sont PC2.0 et PC2.3 sans priorité, les deux seuls algorithmes qui n’utilisent pas de priorité. Ils sont nettement améliorés lorsqu’on leur ajoute une priorité, que ce soit p_1 ou p_2. Ces deux priorités ont un même effet global : elles permettent d’organiser la propagation des feuilles de l’arbre vers la racine. Ceci réduit considérablement le nombre d’étapes de propagation. Un nœud père n’est mis à jour que lorsque tous ses nœuds fils l’ont été, ce qui évite de repasser constamment d’un nœud fils au nœud père.

La différence entre p_2 et p_1 n’est pas nette dans la plupart des cas. Elle est importante surtout dans l’étape 1 de PC2.1 et l’étape 3 de PC2.3. La priorité p_2 apparaît meilleure dans ces deux contextes. Il y a la même idée avec p_1 et p_2 d’organiser la propagation des nœuds fils avant celle de leur père, mais p_2 ajoute en plus un ordre entre nœuds frères. Pour plusieurs nœuds frères, on met à jour en premier le nœud qui a le moins de fils. C’est celui qui subit et exerce le moins de contraintes, donc le moins susceptible d’être modifié par la suite. Et même s’il est modifié, il y aura moins de propagations à refaire que pour un nœud qui aurait plus de fils. On peut donc dans certaines situations très particulières minimiser le nombre de propagations redondantes, ce que l'on ne retrouve que dans quelques étapes du scénario.

La tendance peut même s’inverser légèrement comme dans le cas de PC2.4 si on compare les temps de calcul avec p_1 et p_2 pour la propagation en une seule étape de toutes les informations. En fait, il se peut que l’on soit dans la configuration la plus défavorable pour p_2 et la plus favorable pour p_1. Mais en théorie, avec p_2, on minimise la borne supérieure du temps de calcul.

Après l’introduction d’une priorité, il reste à organiser au mieux les opérations de vérification de cohérence. Dans l’algorithme PC2.1 en effet, on constate que l’on passe beaucoup de temps à faire une vérification de cohérence globale. On vérifie la cohérence pour tous les groupes père/fils alors que parfois aucune des contraintes intervenant dans ce groupe n’a été modifiée depuis la précédente vérification. Toutes ces opérations sont donc inutiles et coûteuses, on essaye donc de les réduire au maximum.

D’où le passage à PC2.2 et PC2.3 avec priorité. Dans ces deux cas, on applique le même principe que pour l'opération de propagation qui consiste à n’appliquer la vérification de cohérence que sur les contraintes modifiées. Pour PC2.2, cette vérification se fait comme auparavant dans une étape bien distincte de la propagation des contraintes. Dans PC2.3 au contraire elle intervient à l’intérieur même de la propagation, dès que la date d’un nœud père ou fils est modifiée.

Cette version est moins efficace car elle crée à nouveau des vérifications inutiles. Si la date d’un nœud fils est modifié, on va immédiatement vérifier la cohérence avant même de propager cette nouvelle contrainte du nœud fils vers le nœud père. Et il faudra recommencer cette vérification si la modification du nœud fils induit une modification de la date du nœud père par propagation.

Enfin, il reste à analyser les deux dernières versions : PC2.4 et PC2.5. Elles sont équivalentes au niveau des résultats. Toutes deux cherchent à généraliser l’idée que les contraintes sont modifiées des feuilles vers la racine de l’arbre, que ce soit par propagation ou par vérification de cohérence. On applique donc aussi un niveau de priorité pour la vérification de cohérence. Ce niveau de priorité est défini comme précédemment et est celui du nœud père. Ceci implique de mêler les opérations de propagation des contraintes et de vérification de cohérence, au gré des niveaux de priorité. Il est inutile de vérifier la cohérence sur le nœud racine si toutes les informations n’ont pas encore été propagées jusque là.

Le gain par rapport à PC2.2 paraît très faible si on prend la somme de toutes les étapes, mais on va gagner du temps sur certaines étapes comme la première par exemple. Ceci signifie qu’avec PC2.2 il y avait des redondances dans le calcul du graphe minimal pour la première étape et ces redondances sont limitées avec PC2.4 et PC2.5. Pour les autres étapes où PC2.2 gagne 1 à 0 secondes sur PC2.4 et PC2.5, on est dans des cas favorables où PC2.2 ne suscite pas de redondance. Dans ces cas là, PC2.4 et PC2.5 ne peuvent pas apporter de gain et il faut leur ajouter le coût (très faible) de gestion des paramètres supplémentaires.

PC2.4 et PC2.5 sont tout de même bénéfiques si on regarde les étapes une par une. Au pire, ils sont très légèrement plus longs, mais dans des cas critiques où PC2.2 entraîne beaucoup d'opérations redondantes, ils peuvent apporter un gain de temps significatif. Ils sont plus efficaces ponctuellement.

La différence entre PC2.4 et PC2.5 est très faible. Dans un cas, on ne gère que la liste des groupes pères/fils où il faut vérifier la cohérence. Dans l'autre, on distingue suivant que c'est une contrainte sur un nœud fils ou sur un nœud père qui a été modifiée en dernier. La vérification de la cohérence n'est pas complète à chaque fois, on n'applique que les étapes nécessaires : étape 1 pour un nœud fils modifié et 2 pour un nœud père modifié. Les résultats sont légèrement meilleurs en décomposant ainsi les différentes propagations.

Si on observe le tableau de résultats sur la base du dernier critère, le temps de propagation de toutes les informations en une seule étape, le classement des différents algorithmes est beaucoup moins net et surtout il est différent (cf. Fig. 66). Il y a tellement d'informations à propager que quasiment toutes les contraintes sont modifiées et l'algorithme est long. Quel que soit l'ordre des opérations choisi, il y a beaucoup de modifications et de retours arrière à effectuer, donc des listes de contraintes à modifier plus lourdes à gérer et un nombre relatif d'opérations redondantes plus petit. Le gain potentiel lié à une meilleure organisation est plus faible alors que le coût de la mise en place d'une telle organisation est plus fort.

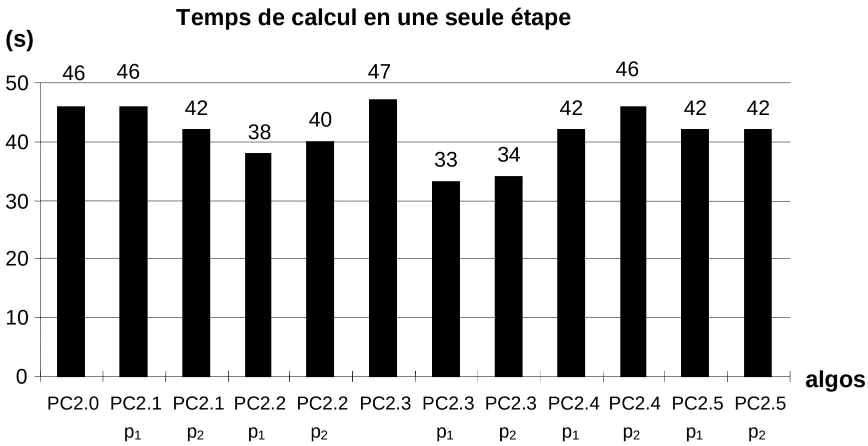


Fig. 66 : comparaison des temps de calcul pour le scénario complet en une seule étape

III.5 Contraintes disjonctives

Les algorithmes utilisés sont valables pour des contraintes floues non disjonctives, ou, ce qui est synonyme, pour des ensembles flous convexes. Ici on les étend à des contraintes disjonctives [Eude 97]. Ceci permet d'élargir le pouvoir d'expression des graphes. On peut introduire des contraintes de la forme "l'événement A arrivera soit dans une heure, soit dans trois heures". Ce type de contrainte traduit une indétermination entre deux dates pour un événement, ce qui ne modifie pas forcément l'activité globale à laquelle cet événement participe.

Si pour une tactique donnée on peut choisir deux types de déroulements très proches à quelques contraintes temporelles près, il peut être intéressant de ne la représenter que par un seul gabarit avec des contraintes disjonctives plutôt que par plusieurs gabarits quasiment égaux. On y gagne en nombre d'hypothèses à gérer, et donc en temps de calcul.

Des contraintes disjonctives peuvent aussi servir à modéliser des "liens multiples" dans la technique des gabarits. A l'intérieur d'un gabarit, une même activité peut se reproduire n fois, mais on ne connaît qu'une fourchette encadrant le nombre n d'occurrences. Ce nombre peut dépendre de l'importance de l'opération par exemple. Suivant son nombre d'occurrences, l'activité n'aura pas la même durée. On peut représenter sa durée par une contrainte disjonctive faisant l'union des durées possibles pour chaque valeur de n autorisée.

Prenons l'exemple d'une activité de décollage. La durée de cette activité dépend fortement du nombre d'avions qui vont décoller. On sait simplement que pour un certain gabarit, il doit y avoir entre 1 et 2 décollages successifs espacés d'une certaine durée. On va modéliser la durée globale de l'activité décollage par une contrainte disjonctive :

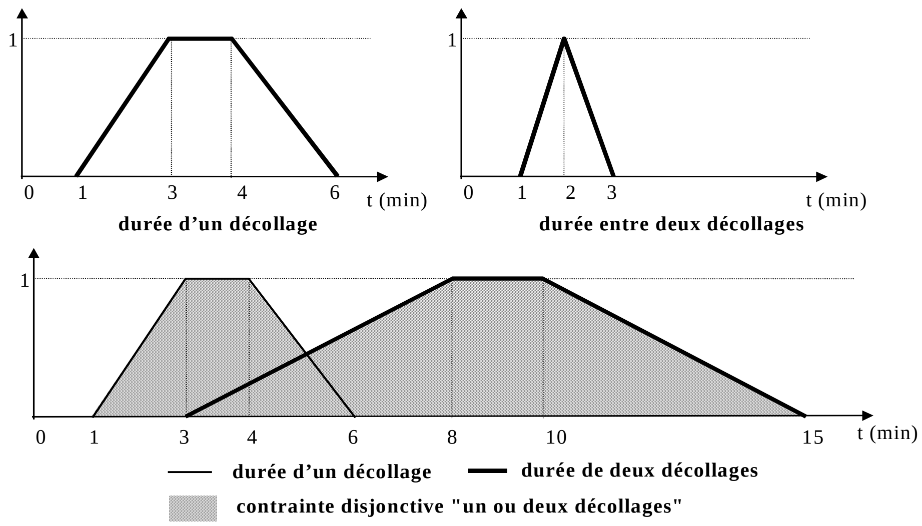


Fig. 67 : exemple d'utilisation d'une contrainte disjonctive

Nous rappelons que les contraintes disjonctives floues sont représentées par des fonctions affines par morceaux. Le calcul des opérations floues d'intersection, de minimum et de somme se fait donc par les algorithmes présentés en III.3.2B)c) p.95.

III.5.1 Consistance locale et minimalité

Un premier théorème de Montanari nous a permis de passer de la 3-consistance à la consistance de chemins simplement en ayant un graphe complet. Il s'applique que les contraintes soient disjonctives ou non. Il est en effet possible de rendre un graphe complet en ajoutant des contraintes universelles partout où il n'y a pas de contrainte. La distinction entre contraintes simples (ou non disjonctives) et disjonctives intervient avec le deuxième théorème de Montanari : si la composition est distributive sur l'intersection, un réseau chemins-consistant est minimal. On en déduit donc que si cette propriété est vérifiée, les algorithmes cités précédemment conduisent directement au graphe minimal.

Ceci s'applique au cas particulier des contraintes numériques avec le théorème de Davis. Dans un CSP numérique, si toutes les contraintes sont des relations d'ordre ou des inégalités de potentiel non disjonctives, l'algorithme de propagation par 3-consistance est complet. Si les inégalités de potentiels sont disjonctives, la distributivité de la composition sur l'intersection n'est plus vérifiée et ce type d'algorithme n'est pas complet : il donne la consistance de chemins mais ne détecte pas forcément l'inconsistance.

Le théorème de Davis s'applique parfaitement aux STP de Dechter [Dechter 91]. Dans le cas des STP, les algorithmes du type PC1 ou PC2 déterminent si le problème est consistant et assurent la minimalité en un temps polynômial. On ne peut pas généraliser les méthodes appliquées aux STP aux TCSP, du fait de la non distributivité de la composition sur l'intersection pour des disjonctions. Dechter montre que dans le cas général, pour des contraintes disjonctives (TCSP), le calcul du graphe minimal est un problème NP-difficile [Dechter 91]. Nous verrons dans le chapitre suivant les méthodes qui permettent de calculer le graphe minimal pour des contraintes disjonctives.

De même pour des contraintes disjonctives floues, la distributivité n'est pas vraie. Un exemple qui le démontre est présenté en annexe VII.6 p.256. Les algorithmes PC1 ou PC2 ne permettent donc pas de calculer le graphe minimal pour des contraintes disjonctives floues.

III.5.2 Traitement des contraintes disjonctives classiques

Dans le cas de contraintes disjonctives classiques (union d'intervalles), plusieurs solutions ont été proposées dans la littérature. Une première méthode, préconisée par Dousson [Dousson 94.2], se contente d'approcher les contraintes disjonctives par des contraintes simples. Pour une contrainte $C=[10,20]\cup[23,28]$ par exemple, il la réduit en $C'=[10,28]$. Il suffit alors d'appliquer les algorithmes de filtrage par consistance de chemins pour obtenir le graphe minimal, et la complexité est celle obtenue pour des contraintes simples. On peut se permettre ce genre d'approximation quand les intervalles de la disjonction sont proches.

Mais dans le cas contraire, on peut modifier la signification de la contrainte en acceptant des valeurs intermédiaires très différentes de celles autorisées à l'origine. Ce traitement n'est pas acceptable pour une contrainte où la disjonction aurait un sens bien marqué. Dire que "Jean viendra demain ou la semaine prochaine" n'a pas du tout la même signification que "Jean viendra n'importe quel jour d'ici la semaine prochaine". Certaines disjonctions sont irréductibles et il faut les traiter en tant que telles.

Un algorithme d'approximation de la consistance de chemins est aussi proposé dans [Schwalb 93]. Cet algorithme repose sur le même principe qui consiste à ne garder pour une contrainte disjonctive que sa borne inférieure et sa borne supérieure. Un algorithme de consistance de chemins est appliqué sur ce nouveau graphe non disjonctif et on réalise ensuite l'intersection avec le graphe disjonctif initial. On répète cette procédure tant que le résultat est modifié. Cette opération permet souvent de réduire les disjonctions et donc d'approcher la consistance de chemins.

Une autre possibilité consiste à séparer le traitement des contraintes disjonctives et des contraintes simples [Fargier 94] ou [Dechter 91]. On considère deux parties dans le graphe : une partie conjonctive qui ne contient aucune contrainte disjonctive et une partie disjonctive. On développe un arbre de recherche où chaque branche correspond à un choix entre A et B pour une contrainte disjonctive de la forme "A ou B". A chaque nœud de l'arbre on effectue une propagation sur la partie conjonctive du graphe.

Cette méthode s'applique très bien dans le cas de l'ordonnancement de production [Fargier 94]. Dans ces problèmes il faut respecter des contraintes de capacité du type "les tâches O_i et O_j ne peuvent pas se dérouler simultanément", et des contraintes de disponibilité comme "O_i ne peut pas débuter avant la date d". Dans l'arbre de recherche développé, chaque nœud correspond à un choix "Oi est avant Oj" qui résout une contrainte de capacité. A chaque nœud correspond donc un nouveau graphe pour lequel on propage toutes les contraintes non disjonctives comme les contraintes de disponibilité.

L'intérêt de l'arbre apparaît surtout lorsque l'on cherche une solution quelconque ou la meilleure solution, mais pas pour une recherche exhaustive de toutes les solutions. Si l'on recherche toutes les solutions en effet, on est obligé de développer la totalité de l'arbre, et la méthode perd son efficacité. Dans ce cas là, la solution adoptée dans [Dechter 91] ou [Fontaine 93] semble plus efficace.

Dans cette solution on décompose chaque contrainte disjonctive en autant de contraintes simples que nécessaire. On construit un graphe pour chaque contrainte simple correspondant à la décomposition, et ainsi de suite sur chaque contrainte. Un TCSP est dérivé en une union de STP comme suit :

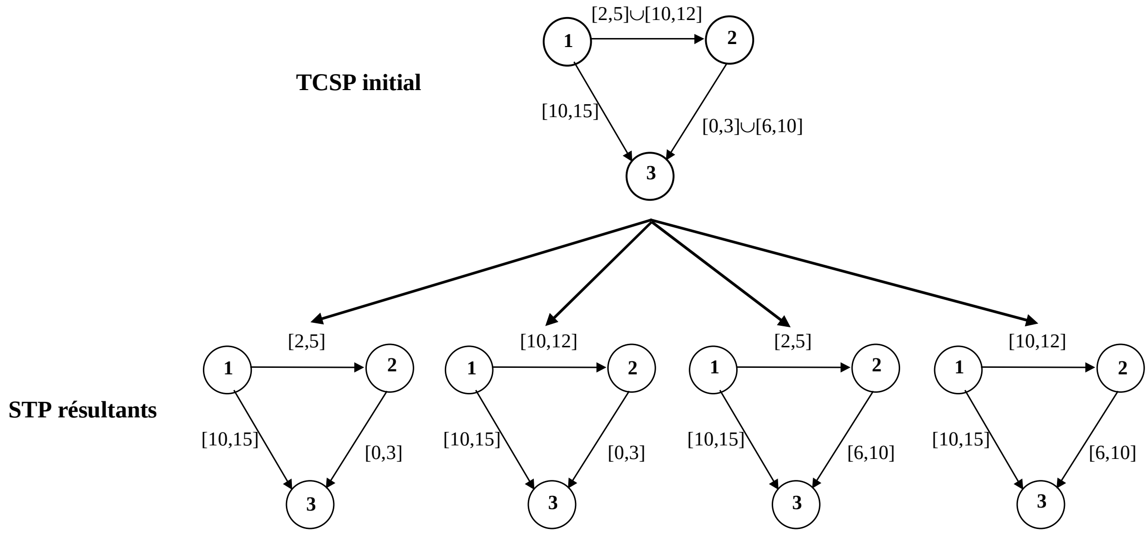


Fig. 68 : décomposition d'un TCSP en STP

On effectue ensuite la propagation sur chaque graphe élémentaire. On obtient les graphes minimaux correspondants. Le graphe minimal global est obtenu en faisant l'union des graphes minimaux.

Par rapport à un STP, la complexité est multipliée par le nombre de graphes élémentaires obtenus par décomposition. Le calcul est en $O(n^3)$ pour un STP. Pour un TCSP ayant un nombre e d'arcs, chaque contrainte se composant au maximum de k intervalles, il y a au maximum k^e graphes minimaux à calculer. La complexité du calcul du graphe minimal pour un TCSP est donc en $O(n^3k^e)$ [Dechter 91].

L'avantage de cette méthode par rapport au développement d'un arbre de recherche complet est que la propagation se fait en une seule fois sur chaque graphe alors qu'elle se fait de façon incrémentale à chaque nœud pour un arbre. Sur l'exemple précédent, on effectue 4 propagations globales. Par la méthode de Fargier, on obtient l'arbre de recherche suivant :

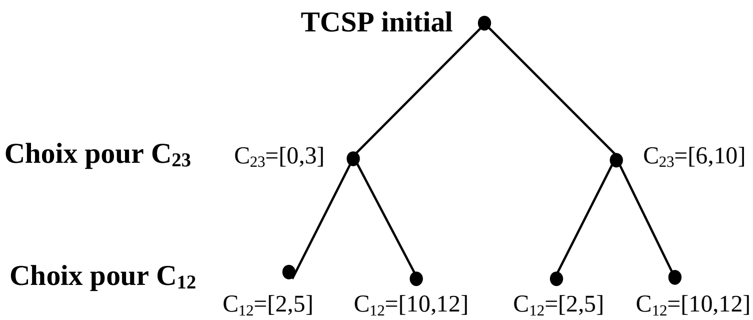


Fig. 69 : résolution d'un TCSP par un arbre de recherche

Il y a 6 nœuds développés, il faut donc faire 6 propagations partielles différentes, sauf si un des graphes devient inconsistant avant d'atteindre une feuille de l'arbre. Chacune de ces propagations est légèrement moins coûteuse qu'une propagation globale, mais on en fait 6 au lieu de 4 avec une méthode de décomposition. L'arbre de recherche est intéressant uniquement s'il y a beaucoup de branches qui sont coupées très tôt.

III.5.3 Traitement des contraintes disjonctives floues

Nous proposons une solution beaucoup plus économique qui reprend en partie l'idée précédente de décomposition. On applique l'algorithme complet dans un premier temps et on obtient un graphe chemins-consistant. Cette étape permet de réduire les contraintes et éventuellement il peut ne plus y avoir de contraintes disjonctives. Dans ce cas très favorable, nous avons démontré la propriété suivante :

Propriété 1 : Si après application d'un filtrage par 3-consistance sur un graphe disjonctif G le graphe obtenu est non disjonctif, alors c'est le graphe minimal équivalent à G.

Démonstration : on peut en effet découper le déroulement de l'algorithme en deux parties. La première partie s'arrête dès que le graphe obtenu G_{int} ne comporte plus de contrainte disjonctive. G_{int} est équivalent à G ($G_{int} \Leftrightarrow G$) puisqu'il est obtenu par propagation à partir de celui-ci. Si on applique les algorithmes de propagation à G_{int}, on obtient son graphe minimal G_{min} puisque l'on est dans le cas de contraintes non disjonctives. $G_{min} \Leftrightarrow G_{int}$ et $G_{int} \Leftrightarrow G$ donc $G_{min} \Leftrightarrow G$. G_{min} est minimal et équivalent à G, c'est donc le graphe minimal de G. ❐

Avant d'envisager toute séparation des contraintes, il peut donc être suffisant d'appliquer les algorithmes de propagation puisque dans certains cas ils assurent la minimalité. Dans ces cas l'algorithme est parfaitement identique à celui appliqué sur des contraintes simples. Dans les autres cas, les algorithmes permettent de réduire le graphe à un réseau chemins-consistant, ce qui nous rapproche déjà du graphe minimal.

S'il reste des contraintes disjonctives, nous montrons que l'on peut appliquer une décomposition en graphes simples comme chez Dechter et Fontaine. La séparation des contraintes disjonctives n'est pas unique pour des contraintes floues, la disjonction étant donnée par l'opérateur maximum non bijectif. Une contrainte disjonctive floue A peut se décomposer en plusieurs combinaisons de contraintes simples A_i et B_i. Les fonctions d'appartenance doivent simplement vérifier :

$$\mu_A(x)=\max[\mu_{A_i}(x),\mu_{B_i}(x)]$$

Voici un exemple de contrainte disjonctive avec deux des séparations possibles en contraintes élémentaires :

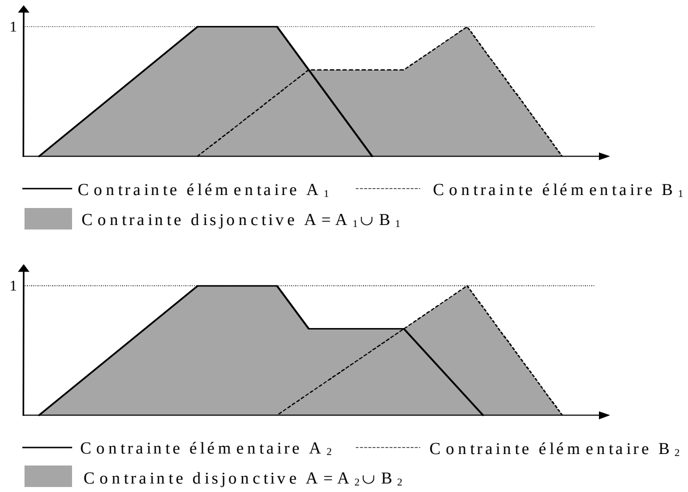


Fig. 70 : séparation d'une contrainte disjonctive floue en contraintes simples

La propriété 2 permet d'affirmer que l'on obtient le graphe minimal par une décomposition en graphes simples quelle que soit la séparation choisie :

Propriété 2 : Si on décompose un graphe disjonctif G en graphes simples G_i tels que toute contrainte C de G de fonction d'appartenance μ est décomposée en C_i et C_j de fonctions d'appartenance μ_i et μ_j avec $\mu(x)=\max(\mu_i(x),\mu_j(x))$ pour tout x, alors le graphe minimal équivalent à G est l'union des graphes minimaux équivalents aux G_i.

Cette propriété se démontre à partir de deux lemmes :

Lemme 1 : G est équivalent à l'union des graphes minimaux équivalents aux G_i.

Démonstration : rappelons quelques définitions indispensables. Au sens classique, deux graphes sont équivalents s'ils ont les mêmes ensembles de solutions. Au sens flou, deux graphes sont équivalents s'ils ont les mêmes solutions avec les mêmes degrés de satisfaction. La solution d'une union de graphes est l'union des solutions. Si deux graphes ont une même solution S_1 avec des degrés de satisfaction différents d_1 et d_2, l'union des graphes a pour solution S_1 avec pour degré de satisfaction $\max(d_1,d_2)$.

Soit G notre graphe initial et C_{ij} ses contraintes (i,j=1 à n). Supposons pour simplifier qu'il ne comporte qu'une seule contrainte disjonctive $C_{i_0j_0}$. On décompose $C_{i_0j_0}$ en deux contraintes C_1 et C_2 et on construit les graphes G_1 et G_2 en remplaçant $C_{i_0j_0}$ respectivement par C_1 et C_2. La seule obligation pour C_1 et C_2 est de vérifier $\mu_{C_{i_0j_0}}(x)=\max[\mu_{C_1}(x),\mu_{C_2}(x)]$. Soit $S=\{(X_0,X_1,...X_n)\}$ une solution de G avec un degré α. Ceci signifie que $\min_{i,j}[\mu_{C_{ij}}(X_i,X_j)]=\alpha$. S est solution de G_k avec le degré d_k :

$$d_k=\min[\min_{i,j\neq(i_0,j_0)}[\mu_{C_{ij}}(X_i,X_j)],\mu_{C_k}(X_{i_0},X_{j_0})],\ k=1,2$$

S est donc solution de $G_1 \cup G_2$ avec le degré d :

$d=\max(d_1,d_2)$

$=\max\{\min[\min_{i,j\neq(i_0,j_0)}[\mu_{C_{ij}}(X_i,X_j)],\mu_{C_1}(X_{i_0},X_{j_0})],$
$\min[\min_{i,j\neq(i_0,j_0)}[\mu_{C_{ij}}(X_i,X_j)],\mu_{C_2}(X_{i_0},X_{j_0})]\}$

$=\max\{\min[A,\mu_{C_1}(X_{i_0},X_{j_0})],\min[A,\mu_{C_2}(X_{i_0},X_{j_0})]\}$ avec $A=\min_{i,j\neq(i_0,j_0)}[\mu_{C_{ij}}(X_i,X_j)]$

$=\min\{A,\max[\mu_{C_1}(X_{i_0},X_{j_0}),\mu_{C_2}(X_{i_0},X_{j_0})]\}$

$=\min[A,\mu_{C_{i_0j_0}}(X_{i_0},X_{j_0})]=\alpha$

Les graphes minimaux G_1^{min} et G_2^{min} équivalents à G_1 et G_2 ont les mêmes ensembles de solutions par définition donc S est aussi solution de $G_1^{min} \cup G_2^{min}$ avec le degré α.

On vient ainsi de montrer que toute solution de G est solution de $\bigcup_i G_i^{min}$ avec le même degré et inversement, donc G est bien équivalent à $\bigcup_i G_i^{min}$ quel que soit le choix de la décomposition. ❐

Lemme 2 : Une union de graphes minimaux $\bigcup_i G_i^{min}$ est un graphe minimal.

Démonstration : par définition, un FCSP est minimal si et seulement si pour toute contrainte C_{ij} et toute valeur x telle que $\mu_{C_{ij}}(x)=\alpha$ alors il existe une solution globale de degré de satisfaction α comportant les valeurs i_0 et j_0 pour les nœuds i et j avec $j_0-i_0=x$. Soit une contrainte C_{ij} de $\bigcup_i G_i^{min}$ et une valeur x_0 telle que $\mu_{C_{ij}}(x_0)=\alpha$. Alors il existe au moins un graphe G_k^{min} comportant la contrainte $C_{k_{ij}}$ telle que $\mu_{C_{k_{ij}}}(x_0)=\alpha$. G_k^{min} étant minimal alors il existe une solution globale S_0 de G_k^{min} de degré α comportant les valeurs i_0 et j_0 pour les nœuds i et j avec $j_0-i_0=x_0$. S_0 est aussi une solution de $\bigcup_i G_i^{min}$ avec un degré au moins égal à α. $\mu_{C_{ij}}(x_0)=\alpha$ donc S_0 est solution de $\bigcup_i G_i^{min}$ avec un degré α. $\bigcup_i G_i^{min}$ est donc bien un graphe minimal. ❐

On peut en conclure que le graphe minimal d'un graphe disjonctif s'obtient en le décomposant d'une façon quelconque en graphes élémentaires non disjonctifs, et en faisant l'union des graphes minimaux associés à ceux-ci. Pour le choix de la décomposition, nous avons opté pour la décomposition la plus simple possible, la simplicité se calculant au nombre de points définissant chaque ensemble flou. Les algorithmes de décomposition sont présentés en annexe VII.7 p.257.

On pense pouvoir gagner du temps en n'effectuant la décomposition qu'après avoir appliqué les algorithmes de propagation de façon à obtenir un graphe chemins-consistant. La décomposition peut alors s'avérer inutile s'il n'y a plus du tout de contrainte disjonctive. Elle sera simplifiée si le nombre de contraintes disjonctives est réduit.

Tous les graphes simples obtenus par décomposition ont en effet une partie commune, la partie non disjonctive du graphe initial. Il est inutile de propager les contraintes sur cette partie puisque cela a déjà été fait dans la première étape. On ne propage que les contraintes disjonctives qui ont été séparées ou celles qui sont modifiées. On gagne donc en temps de calcul par rapport à Dechter et Fontaine qui effectuent les mêmes propagations sur cette partie non disjonctive pour chaque graphe simple.

III.5.4 Résultats

On a comparé de façon expérimentale les deux méthodes qui s'offrent à nous en cas de contraintes disjonctives, celle que l'on vient de présenter et celle de Dechter. Dans le deuxième cas on construit tout de suite des graphes simples en séparant les contraintes disjonctives initiales. Voici une présentation du premier cas sous une forme algorithmique :

Algorithme de traitement des contraintes disjonctives :

- 1. Appliquer PC2 sur le graphe disjonctif G_1
- 2. Si le graphe obtenu G_2 est disjonctif faire :
- 3. Décomposer G_2 en graphes simples G_{2_i}
- 4. Appliquer PC2 sur les G_{2_i} pour obtenir les $G_{2_i}^{min}$
- 5. La solution est $\cup(G_{2_i}^{min})$
- 6. Sinon la solution est G_2

Qualitativement, les résultats obtenus par ces deux méthodes sont équivalents. Les seules différences sont dues aux arrondis et sont de l'ordre de la précision de calcul requise. Le critère de comparaison important est le temps de calcul. Mais le résultat de cette comparaison peut varier suivant la nature du graphe que l'on minimise. Nous allons détailler la comparaison pour différents cas de graphes initiaux.

Les graphes initiaux sont construits pour les besoins de l'expérimentation : on fait varier leur taille, le nombre de leurs contraintes disjonctives et le résultat de la propagation par consistance de chemins (consistance, inconsistance, présence ou non de contraintes disjonctives après propagation). Tous les exemples de graphes présentés maintenant sont réalistes mais ne peuvent pas être caractérisés a priori.

L'algorithme appliqué est la version optimale obtenue, PC2.5. On n'impose aucune contrainte structurelle, donc aucune vérification de cohérence père/fils. Il s'agit de tester la propagation sur des contraintes disjonctives sans autre considération hiérarchique. Les temps de calcul fournis sont des temps moyens.

A) Cas 1

Le premier cas regroupe des graphes consistants ne comportant plus de contrainte disjonctive après propagation. C'est a priori le cas le plus favorable pour notre méthode (méthode 1). On effectue en effet une première propagation et le résultat ne comporte plus de contrainte disjonctive donc il n'y a pas de séparation de contraintes à faire. Dans la méthode de Dechter (méthode 2) on sépare directement les contraintes disjonctives et donc on propage sur tous les graphes simples. Le nombre de graphes simples dépend du nombre de disjonctions.

Voici un tableau récapitulatif des résultats obtenus :

graphes	caractéristiques	méthode 1	méthode 2
1	3 nœuds 1 disjonction	50 ms	65 ms
2	3 nœuds 2 disjonctions	55 ms	130 ms
3	7 nœuds 1 disjonction	sans priorité : 1s 420ms avec priorité : 570 ms	700 ms 630 ms
4	7 nœuds 2 disjonctions	sans priorité : 1s 460ms avec priorité : 580 ms	930 ms 750 ms

Sur le graphe 1, les temps de calcul sont très faibles : on passe de 50 millisecondes avec la méthode 1 à 65 ms avec la méthode 2. La différence n'est pas très importante du fait des temps de calcul faibles. Elle confirme simplement l'avantage de la méthode 1 par rapport à la méthode 2 dans ce cas de figure. On calcule un graphe minimal au lieu de deux, mais la différence en temps de calcul n'est pas du simple au double. Les calculs sont en effet plus longs avec une contrainte disjonctive puisqu'elle est représentée par un plus grand nombre de points. Le nombre de graphes à calculer est en partie compensé par la simplicité des calculs pour chacun de ces graphes.

La différence est accentuée si on augmente le nombre de contraintes disjonctives. Dans le graphe 2 on remplace une contrainte simple du graphe 1 par une contrainte disjonctive qui sera elle aussi réduite. Les temps de calcul vont de 55 à 130 ms. Par la méthode 1, l'ajout de la disjonction n'augmente que très faiblement le temps de calcul, alors que par la méthode 2 il le double : il faut calculer 4 graphes minimaux au lieu de 2.

On a ensuite augmenté la taille du graphe 1 en lui ajoutant 4 nœuds et des contraintes simples. Les contraintes ajoutées sont déjà minimales, elles ne modifient pas le résultat de la propagation. Les temps de calcul pour le graphe 3 sont alors inversés : 1 s 420 ms pour la méthode 1 et 700 ms pour la méthode 2. Ceci vient de l'ordre des opérations. Dans la méthode 1 en effet, la contrainte disjonctive peut être réduite en une contrainte simple dès le début ou tout à la fin suivant l'ordre des propagations. Dans le cas présent où elle n'est pas réduite, elle est propagée telle quelle et toutes les opérations qui l'utilisent doivent calculer 6 points au lieu de 3. On peut la forcer à être réduite tout de suite en la rendant prioritaire.

Si on augmente la priorité du nœud qui la concerne, elle est mise à jour et donc réduite en premier et on diminue fortement le temps de calcul. On passe en effet à 570 ms au lieu de 1 s 420 ms précédemment. Théoriquement, l'ajout de cette priorité ne doit pas influer sur la méthode 2 : la légère diminution du temps de calcul (630 ms) est due au hasard des simplifications de calcul et n'est pas contrôlable.

Si on emploie la méthode 1, on a donc intérêt à rendre les contraintes disjonctives prioritaires. Si l'on a déjà une priorité liée à la structure hiérarchique des gabarits, on rend les contraintes disjonctives prioritaires seulement par rapport aux contraintes de même niveau hiérarchique. On doit garder l'ordre de propagation par niveau dans l'arbre, on ne peut modifier l'ordre qu'à l'intérieur d'un niveau.

Le graphe 4 est obtenu soit en ajoutant une contrainte disjonctive au graphe 3, soit en augmentant la taille du graphe 2. On effectue les mêmes constats que précédemment : il est d'autant plus nécessaire d'introduire une priorité pour les contraintes disjonctives que le graphe est grand, et la méthode 1 est d'autant plus avantageuse par rapport à la méthode 2 que le nombre de contraintes disjonctives est important.

Si l'on compare les temps de calcul par la méthode 2 pour les graphes 3 et 4, on passe seulement de 700 à 930 ms alors que l'on double le nombre de graphes à calculer. On calcule les deux mêmes graphes plus deux graphes supplémentaires qui sont eux inconsistants ; ils correspondent à la partie disjonctive que l'on a ajoutée et qui n'intervient pas dans la solution. Le calcul de ces graphes est beaucoup plus court si on détecte tout de suite l'inconsistance.

B) Cas 2

Nous allons maintenant analyser le cas de graphes inconsistants. L'inconsistance peut être détectée dès la première propagation. Il n'y a plus de contrainte disjonctive après cette étape et la séparation des contraintes est donc inutile. Les tests sont plus rapides et ne sont pas significatifs pour des graphes de 3 nœuds comme précédemment.

On s'intéresse uniquement à des graphes de 7 nœuds dont voici les caractéristiques :

graphes	caractéristiques	méthode 1	méthode 2
5	7 nœuds 1 disjonction	sans priorité : 50 ms avec priorité : 90 ms	130 ms 120 ms
6	7 nœuds 2 disjonctions	sans priorité : 50ms avec priorité : 100 ms	300 ms 250 ms

Le graphe 5 montre bien qu'il y a une grosse différence entre la méthode 1 et la méthode 2. La méthode 1 est encore favorable dans ce cas puisque tout le calcul de la séparation des contraintes est inutile. Mais cette fois-ci l'ajout d'une priorité pour la contrainte disjonctive est pénalisante. Dans l'exemple particulier du graphe 5, l'incohérence ne porte pas sur la disjonction, et si on met à jour la contrainte disjonctive en priorité on retarde la détection de l'inconsistance. La méthode 1 reste néanmoins avantageuse.

Ceci est encore renforcé avec le graphe 6 qui oblige à calculer 4 graphes par la méthode 2 au lieu d'un seul par la méthode 1. La priorité est à nouveau pénalisante mais on ne peut pas le savoir a priori et donc il vaut mieux l'utiliser dans tous les cas. Dans un cas favorable elle permet un gain de temps beaucoup plus important que ce qu'elle fait perdre dans un cas où elle est pénalisante.

C) Cas 3

On regarde maintenant des cas défavorables où la séparation des contraintes est obligatoire. Les graphes sont tous déjà chemins-consistants et la première propagation ne modifie aucune contrainte. Les contraintes disjonctives sont intactes et il faut les séparer et refaire la propagation sur chaque graphe simple.

Voici les résultats obtenus :

graphes	caractéristiques	méthode 1	méthode 2
7	3 nœuds 2 disjonctions	sans priorité : 265 ms avec priorité : 265 ms	140 ms 140 ms
8	7 nœuds 2 disjonctions	sans priorité : 670ms avec priorité : 660 ms	1 s 50 ms 900 ms

Pour le graphe 7, l'ajout d'une priorité ne change rien : le graphe est trop petit pour que l'ordre des opérations joue un rôle quelconque. La méthode 2 s'avère cette fois-ci beaucoup plus efficace que la première. Ceci paraît normal puisque le premier calcul de propagation est a priori totalement inutile. En fait, il ne l'est pas, et nous allons le constater sur un plus gros graphe comme le graphe 8.

La tendance s'inverse en effet, et la méthode 1 est plus efficace. Pendant la première propagation de la méthode 1, on vérifie que tous les chemins sont consistants. Après la séparation des contraintes, on ne vérifie pas à nouveau tous les chemins mais seulement ceux pour lesquels des contraintes ont été modifiées. Pour chaque graphe simple, seules 2 contraintes sont modifiées puisqu'il y a 2 contraintes disjonctives, sur un total de 11 contraintes. Une partie des vérifications a déjà été effectuée pendant la première propagation, commune à tous les graphes, et on ne refait pour chaque graphe simple qu'un minimum d'opérations.

Si on n'effectue pas cette première propagation en utilisant la méthode 2, toutes les vérifications sont à faire pour chaque graphe simple et donc on fait plusieurs fois les mêmes opérations de vérification, ce qui constitue finalement une perte de temps pour un graphe de taille supérieure à 3 ou 4 nœuds.

D) Limitations

Le traitement des contraintes disjonctives connaît tout de même des limitations indépendantes du choix de la méthode. Le nombre de contraintes disjonctives fait croître très rapidement le nombre de graphes simples à calculer. Si on a n contraintes disjonctives, chacune se décomposant en deux contraintes simples, il y a 2^n graphes à calculer, ce qui devient rapidement ingérable. On peut effectuer une première propagation sans séparer les contraintes en espérant que les contraintes disjonctives se simplifient. Mais d'une part on n'est pas assuré de cette simplification, et d'autre part les calculs sur les contraintes disjonctives sont plus coûteux puisqu'il y a plus de points à calculer.

Dans certains cas extrêmes, une première propagation peut même faire augmenter le nombre de disjonctions. Sur un exemple de graphe à 5 nœuds, on a au départ 6 contraintes disjonctives se séparant chacune en 2 contraintes simples. Si on effectue tout de suite la séparation des contraintes, cela fait 2^6 donc 64 graphes minimaux à calculer. Si on propage d'abord, avec l'exemple particulier de graphe construit pour ce test, les domaines se réduisent et les contraintes disjonctives sont composées non plus de 2 mais de 4 contraintes simples chacune. On a par exemple des noyaux de contraintes qui sont $[1,4]\vee[6,9]$, et par propagation certaines valeurs se trouvent exclues et les noyaux deviennent $[1,2]\vee[3,4]\vee[6,7]\vee[8,9]$. On a alors 4^6 donc 4096 graphes minimaux à calculer, ce qui devient irréalisable. Quelle que soit la méthode employée, il faut limiter au maximum l'emploi de contraintes disjonctives.

Nous avons testé notre algorithme de traitement des contraintes disjonctives sur des exemples factices. Ces exemples ont été construits de façon à couvrir un maximum de situations possibles, des cas les plus favorables aux cas les plus défavorables. Il reste à le tester sur des expérimentations avec des gabarits réels, ce qui nécessite d'avoir un dispositif de simulation important, comportant des gabarits extraits de l'expertise et un scénario complet réaliste.

III.6 Pouvoir d’expression

L’objectif ici est de recenser toutes les expressions temporelles que l’on peut exprimer sous la forme de graphes de contraintes temporelles floues et de voir la richesse supplémentaire qu’un tel formalisme apporterait à la technique des gabarits. Le flou permettra essentiellement d’apporter une certaine notion d’imprécision et surtout de satisfaction partielle par rapport aux expressions temporelles habituelles.

Pour exprimer la durée d’un intervalle, on est obligé de le décomposer en sa date de début et sa date de fin. Le besoin de représenter des durées d’activités nous contraint donc à considérer le point comme élément temporel de base. La gestion simultanée d’éléments ponctuels et durables est un peu lourde et on se limite donc aux instants, qui permettront déjà une richesse d’expression suffisante. On peut modéliser toutes les contraintes numériques habituelles ainsi que des contraintes de dates ou de durées floues.

Le pouvoir d'expression dépend de l'utilisation ou non de contraintes disjonctives, puisque l'on a vu précédemment que les disjonctions peuvent permettre de réduire le nombre de gabarits à considérer mais sont aussi plus complexes. Considérons d'abord le cas où les disjonctions sont exclues. On pourra cependant représenter toutes les relations symboliques exprimables sans disjonction par des relations entre points. Ceci comprend entre autres les relations symboliques de l’algèbre d’instants sauf la relation $\neq$ et toutes les primitives d’Allen. On peut de même traduire les expressions symboliques comme "aujourd’hui, maintenant"… ainsi que les modificateurs linguistiques "beaucoup, très"…

La construction des différents ensembles flous correspondant à ces expressions peut être obtenue de façon automatique par une grammaire comme présenté sur l'exemple du chapitre II.1.4C) p.28. Mais ils peuvent aussi être établis d’après le recueil d’expertise, lors de la construction des gabarits. Les notions temporelles dépendent en effet fortement du contexte, et une même expression n’a pas toujours la même signification.

Si l’on dit que "deux coureurs de 100 mètres sont arrivés en même temps" ou que "deux avions de ligne arrivent en même temps à l’aéroport" (point de vue de la personne qui attend des passagers), "en même temps" n’a pas la même valeur temporelle. Cette expression peut avoir une précision d’un centième de seconde dans le premier cas et de quelques minutes dans le second. Une grammaire temporelle ne pourrait donc pas être valable dans plusieurs contextes différents, ou alors il faudrait introduire une notion de granularité variable et construire les ensembles flous en fonction d'un niveau de granularité qui serait ensuite adapté au contexte.

Si l'on autorise les contraintes disjonctives, on augmente encore le pouvoir d'expression. On pourra en effet représenter toutes les relations symboliques de l'algèbre d'instants et toutes les relations de l'algèbre d'intervalles restreinte. Il est difficile de caractériser les relations qui appartiennent à cette algèbre mais on pourra représenter un bon nombre des disjonctions de primitives d'Allen.

L'apport de la représentation par graphes temporels ne concerne pas seulement l'expression des contraintes temporelles en elles-mêmes mais aussi les éléments que l'on peut contraindre. Dans la technique des gabarits, on ne pouvait contraindre temporellement que des sous-activités liées par la structure : sous-activités sœurs ou mère / fille. Avec une traduction sous la forme de graphes temporels, on obtient beaucoup plus de souplesse puisque l'on peut contraindre n'importe quelles activités sans considérer la structure. Ainsi on pourra imposer des contraintes temporelles entre des sous-activités qui ne participent pas à une même activité globale.

Les aspects temporels sont ainsi quasiment décorrélés des aspects structurels. Les seules informations présentes dans la structure que l'on conserve pour le raisonnement temporel sont les informations concernant la filiation des activités afin de vérifier toujours les contraintes implicites de cohérence père / fils et les niveaux de priorité liés à la structure.

III.7 Conclusion

L'originalité des travaux que nous présentons réside dans la prise en compte de notions temporelles floues pour la reconnaissance. Dans tout ce qui était fait jusqu'ici, l'imprécision des données temporelles était marquée uniquement par l'utilisation d'intervalles numériques. L'introduction de contraintes floues va surtout permettre d'évaluer beaucoup plus finement les hypothèses de scenarii et par conséquent d'ajouter de la souplesse dans la reconnaissance. La représentation par des graphes permet aussi d'améliorer le pouvoir d'expression temporel en autorisant des délais entre n'importe quelles activités.

L'introduction de contraintes floues a conduit à étendre les algorithmes classiques de recherche du graphe minimal au cas flou. Les bases théoriques étaient déjà présentes avec les FCSP, mais comme on l'a vu, l'application pratique n'est pas immédiate. Les FCSP avaient en effet été envisagés jusqu'ici dans l'optique de la recherche d'une solution particulière mais pas pour le calcul de fenêtres temporelles minimales. Il faut aussi adapter les algorithmes au cas particulier d'une reconnaissance de scenarii temporels hiérarchisés. La structure initiale hiérarchique des gabarits induit en effet des modifications dans les algorithmes.

L'étude des FCSP permet de conclure sur certains choix à faire pour le calcul d'un graphe minimal. Le flou induit une complexité supplémentaire et il faut faire un compromis entre complexité et précision. Si l'on souhaite une précision d'environ 0,1 sur les degrés d'appartenance, on choisira soit une modélisation par 10 α-coupes, soit une modélisation par fonctions affines par morceaux. La première solution entraîne des calculs plus courts, mais la seconde permet une représentation spécifique des contraintes disjonctives.

L'algorithme choisi est du type Montanari (approche CSP). C'est une adaptation aux contraintes floues de PC2. On ajoute en plus des heuristiques afin d'améliorer en pratique le calcul du graphe minimal. Une notion de priorité permet d'ordonner les calculs et on traite les vérifications de cohérence comme des contraintes qui sont propagées au même titre que les autres. On préconise donc l'utilisation de PC2.5, légèrement plus rapide que PC2.4. Une priorité de type p_2 semble la meilleure si on veut minimiser le temps de calcul au pire cas.

Les temps de calcul peuvent paraître très importants par rapport à ceux obtenus avec des contraintes classiques. Mais on peut tenir compte d'une information avant d'effectuer la propagation. Si un événement est daté par t_1 et que t_1 est dans la fenêtre temporelle acceptable de l'événement avec un degré α, on peut dire immédiatement que l'hypothèse vérifie ses contraintes temporelles avec un degré α. Sinon on peut exclure le gabarit. La propagation sert à calculer les nouvelles fenêtres temporelles des événements. L'évaluation ne pourra s'effectuer qu'une fois la propagation terminée, mais dans un premier temps on peut déjà connaître les gabarits retenus.

On a de plus amélioré le calcul du graphe minimal dans le cas de contraintes disjonctives. Le choix de ce type de contraintes impose une modélisation particulière des ensembles flous sous forme de fonctions linéaires par morceaux, plus coûteuse que la modélisation par α-coupes. Il faut là aussi faire un compromis : interdire les contraintes disjonctives réduit la complexité pour le calcul du graphe minimal mais impose d'avoir un plus grand nombre de modèles de gabarits, donc plus d'hypothèses à gérer.

Pour ce qui est de la limitation du nombre de contraintes disjonctives, elle n'est pas gênante pour l'application aux gabarits. Les disjonctions n'interviennent que rarement, dans le cas où le gabarit comporte plusieurs variantes. Les disjonctions ne portent que sur quelques éléments et sont souvent résolues avec peu d'indices : seulement quelques informations sur des événements datés permettent de discriminer les options proposées par les disjonctions, ce qui permet d'éliminer ces disjonctions et de ne retenir qu'une seule hypothèse de déroulement de l'activité.

IV Evaluation des hypothèses

Nous avons vu comment raisonner avec des contraintes temporelles imprécises dans la reconnaissance d'activités militaires. L'intérêt des contraintes floues est d'introduire des nuances dans l'évaluation de leur satisfaction. Il faut donc modifier le mécanisme d'évaluation des différentes hypothèses de gabarits. Le système doit tenir compte de la satisfaction des contraintes temporelles en plus de tous les paramètres liés à la structure du gabarit.

On considère pour l'instant un gabarit qui ne possède que des contraintes temporelles. Un gabarit de ce type est totalement reconnu si l'on a observé toutes les activités qui le composent nécessairement (on peut ne pas avoir observé certaines activités facultatives) et si ces activités vérifient toutes les contraintes temporelles du gabarit. Le degré de reconnaissance d'un gabarit dépend donc de la complétude des observations et de la satisfaction des contraintes.

On ne prend pas en compte ces paramètres structurels avec une mesure de compatibilité du type de celle utilisée dans [Fontaine 98]. Ici en effet le degré de proximité calculé est un degré de compatibilité global ne prenant en compte que les aspects temporels. Ce degré de proximité calcule en fait la probabilité qu'une solution de la session soit aussi solution du scénario.

Nous allons voir comment on peut décomposer l'évaluation en deux étapes, la première mesurant la compatibilité d'observations avec un gabarit donné et la deuxième calculant un degré de reconnaissance global pour le gabarit à partir de ces compatibilités partielles. La compatibilité partielle fait intervenir des mesures de comparaison. Nous présentons donc les exigences vis à vis de cette mesure et les choix possibles parmi les mesures de comparaison existantes. La deuxième étape est constituée d'agrégations successives. Le choix d'une mesure d'agrégation particulière dépend de l'interprétation que l'on en fait. Nous allons donc présenter plusieurs mesures d'agrégation utilisables dans l'évaluation des gabarits avec leur interprétation particulière dans ce cadre.

IV.1 Décomposition de l'évaluation

Un gabarit se présente sous la forme d'une décomposition en sous-activités. Chaque lien de décomposition peut avoir une signification particulière. Il peut être une conjonction, une disjonction ou autre. Nous détaillerons les différentes interprétations possibles dans la partie agrégation (IV.3 p.149). Chaque sous-activité a une importance suivant qu'elle est plus ou moins caractéristique de l'activité globale modélisée par le gabarit. On attache aussi à chaque sous-activité des contraintes temporelles floues de durée ou de délai par rapport à une autre activité. Un exemple de gabarit avec tous ces éléments est présenté sur la figure Fig. 87 p.149.

On distingue deux catégories de sous-activités. Les sous-activités élémentaires qui sont les feuilles des gabarits : elles peuvent être reconnues directement d'après l'observation. Des exemples de telles activités sont une activité ***reconnaissance météo***, un ***entraînement de base***, un ***décollage***… Les autres sous-activités sont elles-mêmes décomposées en sous-activités et ne peuvent être reconnues qu'à travers leurs sous-activités composantes. Ce sont des activités composites. Un gabarit est un cas particulier d'activité composite.

Une sous-activité élémentaire est reconnue si l'on a observé une activité de même nature qui puisse vérifier les contraintes temporelles spécifiées dans le gabarit. La compatibilité entre la nature de l'activité observée et celle de l'activité spécifiée par le gabarit est binaire : les activités sont de même nature ou sont différentes. Les contraintes temporelles spécifiées par le gabarit sont floues, et les dates ainsi que les durées des observations peuvent aussi être imprécises donc floues. Il faut donc établir un degré de compatibilité entre chaque contrainte floue issue de l'observation et la contrainte correspondante du gabarit. De là on peut déduire une compatibilité de la sous-activité.

Une sous-activité composite est reconnue avec un degré qui dépend de la compatibilité partielle de ses sous-activités composantes, de leur importance et de l'agrégation de celles-ci. On distingue donc deux mesures de compatibilité, la compatibilité des observations avec une activité élémentaire et la compatibilité des sous-activités élémentaires partiellement reconnues avec une activité composite. On peut représenter l'évaluation de façon schématique sur un gabarit composé de deux sous-activités élémentaires B et C et d'une activité composite A :

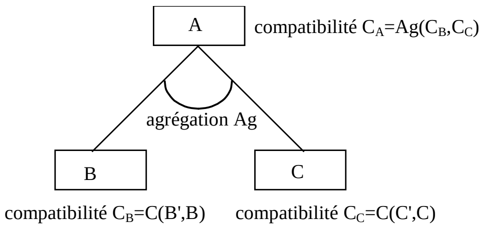


Fig. 71 : décomposition de l'évaluation d'un gabarit

Le gabarit A se décompose en deux activités B et C. La nature de cette décomposition (disjonctive, conjonctive ou autre) est donnée par le type d'agrégation Ag. On a observé un événement B' de la nature de l'activité B et un événement C' de la nature de l'activité C. L'évaluation globale du gabarit A se fait en deux étapes : dans un premier temps on évalue les compatibilités de B' avec B et de C' avec C et ensuite on agrège ces compatibilités partielles en une compatibilité globale C_A=Ag(C_B,C_C). Si aucun événement du type de B n'a été observé, la compatibilité partielle en B est nulle et c'est en ce sens que l'on tient compte de la complétude des observations par rapport au gabarit.

Pour n'importe quel gabarit on retrouve ces deux étapes : évaluation des compatibilités partielles entre les événements et chaque nœud puis agrégation des compatibilités partielles des nœuds à chaque niveau hiérarchique. Pour un gabarit comportant plus de deux niveaux, la compatibilité de certaines sous-activités composites est obtenue par agrégation de compatibilités elles-mêmes issues d'agrégations.

La compatibilité partielle tient compte de la complétude (si l'événement est non observé, la compatibilité pour ce nœud est nulle) et doit aussi tenir compte de la manière dont les contraintes sont satisfaites. L'agrégation dépend de la nature de la composition et des compatibilités partielles des éléments à agréger. Le mode d'agrégation peut être plus ou moins pessimiste ou optimiste, chaque nœud est plus ou moins significatif pour la reconnaissance.

Pour le calcul des compatibilités partielles au niveau de sous-activités élémentaires il faut introduire une mesure de comparaison entre une contrainte temporelle floue observée et une autre à respecter. Pour le calcul des compatibilités au niveau de sous-activités composites on introduit plusieurs mesures d'agrégation avec des significations différentes.

IV.2 Mesures de compatibilité

IV.2.1 Que faut-il comparer ?

L'évaluation d'un gabarit repose sur la comparaison entre des activités observées et le modèle du gabarit. Ce modèle est constamment mis à jour avec les observations supposées lui correspondre. La mise à jour se fait en intégrant et en propageant les nouvelles contraintes, ce qui correspond au calcul du graphe minimal du scénario actualisé. On propage aussi les informations issues de l'observation afin d'assurer leur cohérence et d'en déduire un maximum de renseignements. Les fenêtres temporelles des activités observées sont données par le graphe minimal de la session.

On dispose donc d'un côté, des informations sur certaines activités observées avec des contraintes temporelles réelles (dates, durées ou délais par rapport à d'autres activités) et d'un autre côté, des informations sur le déroulement théorique des activités pour satisfaire le modèle. Ce sont ces deux types d'informations qu'il faut comparer pour évaluer la compatibilité entre la réalité et le modèle.

Dans cette partie nous allons évaluer la compatibilité entre un événement B' et une activité élémentaire B d'un modèle de gabarit. La compatibilité est non nulle si B' est de la même nature d'activité que B. Si c'est le cas, B' doit vérifier les contraintes imposées à B et la compatibilité dépend de la possibilité ou non pour B' de satisfaire les contraintes imposées à B. Ces contraintes sont du type durée ou délai par rapport à une autre activité.

Les observations peuvent être incomplètes et imprécises : B' dure entre 10 et 20 minutes mais on ne sait dater ni son début ni sa fin par exemple. Dans ce cas le graphe de la session donne la contrainte unitaire pour les dates de début et de fin de B'. De plus les contraintes à vérifier sont floues, c'est-à-dire qu'elles peuvent être plus ou moins satisfaites.

Soit C une contrainte imprécise à satisfaire liée à l'activité B (la durée de B par exemple). On observe deux événements B'_1 et B'_2 pouvant tous deux être compatibles avec B. B'_1 vérifie une contrainte C_1 (durée de B'_1) et B'_2 une contrainte C_2 (durée de B'_2) comme représentées sur la figure ci-dessous :

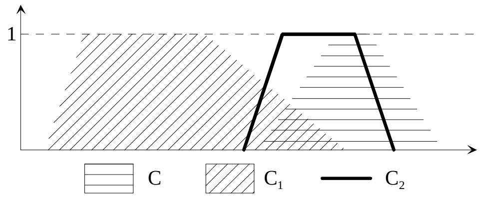


Fig. 72 : compatibilité de contraintes

D'après cette figure, C_2 a beaucoup plus de chance de satisfaire C que C_1. Les durées réelles de B'_1 et B'_2 sont en effet dans C_1 et C_2 respectivement, et on peut remarquer qu'il y a peu de valeurs dans C_1 qui satisfont la contrainte C et aucune qui la satisfait totalement, alors que quasiment toutes les valeurs dans C_2 satisfont C. On aimerait pouvoir dire que B'_2 est plus compatible que B'_1 avec B et mesurer à quel degré.

Rappelons que pour une contrainte floue C de fonction d'appartenance μ_C, $\mu_C(x)$ représente le degré de satisfaction de la contrainte en x. Si la connaissance de B' donnait une valeur précise pour la variable x, on pourrait donc donner un degré de satisfaction de la contrainte C portant sur B. Dans notre cas, B' est incomplètement datée et on ne connaît qu'une estimation de x donnée par une valeur floue X. On appelle X la contrainte réelle (contrainte vérifiée de façon certaine par B'). On ne peut pas dire si X satisfait C mais en revanche on peut estimer ses chances de satisfaire la contrainte en comparant X et C. On appellera ici cette comparaison une mesure de compatibilité puisqu'il s'agit de vérifier qu'une valeur floue est compatible avec une contrainte floue.

On cherche en fait à mesurer la compatibilité entre les faits réellement observés et un modèle d'activité partiellement instancié. Les contraintes du modèle partiellement instancié sont celles du graphe du scénario actualisé. Pour instancier le modèle, on a fait dans un premier temps l'hypothèse de la compatibilité des faits avec le modèle. Si le graphe du scénario actualisé est cohérent, cela signifie que la compatibilité est possible. Lors de la propagation des contraintes, il se peut que les contraintes deviennent non normalisées. Toutes les contraintes du graphe sont ramenées à la même hauteur. Cette hauteur est le degré de satisfaction global des contraintes, il correspond au meilleur degré de satisfaction possible de toutes les contraintes.

Ce degré est une estimation optimiste du degré de compatibilité, mais on souhaite affiner cette estimation. Un même degré de satisfaction global peut en effet correspondre à des situations très différentes. Supposons que l'on n'ait qu'une seule contrainte à vérifier, une durée comprise entre 10 et 20 minutes. Dans un premier cas, les observations induisent que la durée réelle est comprise entre 10 et 15 minutes. Le degré de satisfaction global est alors de 1 et on est sûr de vérifier totalement la contrainte puisque quelle que soit sa valeur entre 10 et 15 minutes, elle respectera toujours la contrainte initiale [10,20].

Dans un deuxième cas, les observations induisent une durée réelle comprise entre 15 et 60 minutes. Le degré de satisfaction global est aussi de 1 puisque si la durée est effectivement de 15 minutes la contrainte sera parfaitement satisfaite. Mais la durée réelle peut aussi prendre toutes les valeurs de 20 à 60 minutes, auquel cas la contrainte ne serait pas du tout vérifiée.

On voit donc que les deux cas sont identiques du point de vue du degré de satisfaction global des contraintes mais peuvent évoluer de façon opposée. Il faut donc affiner ce degré de satisfaction global issu de la comparaison entre les contraintes du graphe de la session et celles du graphe du scénario actualisé.

On aura un degré de compatibilité pour chaque contrainte liée au nœud. Le degré de compatibilité du nœud s'obtient en combinant les degrés de compatibilité de chaque contrainte portant sur ce nœud. On discutera du mode de combinaison par la suite, ici on s'intéresse uniquement au calcul de la compatibilité pour une contrainte.

IV.2.2 Différentes mesures de compatibilité

Nous allons voir ici quelles sont les mesures sur les ensembles flous proposées dans la littérature et pouvant être interprétées comme une mesure de compatibilité.

A) Possibilité et nécessité

Plusieurs mesures de compatibilité de la littérature sont fondées sur les notions de possibilité et nécessité relatives. Soit A le sous ensemble flou représentant la référence et A' le sous ensemble flou représentant l'objet à comparer à A. La possibilité de A relative à A' est [Dubois 88.2] :

$$\Pi(A, A') = \sup_{x \in X} \min(\mu_A(x), \mu_{A'}(x))$$

Elle dénote la possibilité que A' corresponde à A. Elle peut être vue comme un degré d'intersection [Rossazza 90].

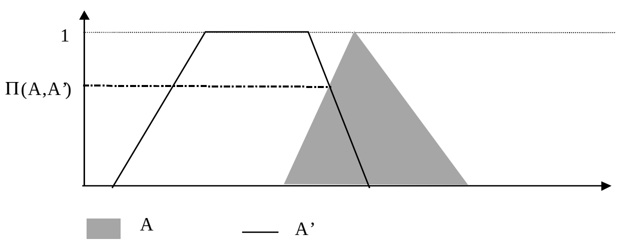


Fig. 73 : possibilité de A relative à A'

La mesure duale de la possibilité relative est la nécessité de A relative à A' :

$$N(A, A') = 1 - \Pi(\neg A, A') = \inf_{x \in X} \max(\mu_A(x), 1 - \pi_{A'}(x))$$

Elle dénote la nécessité que A' corresponde à A.

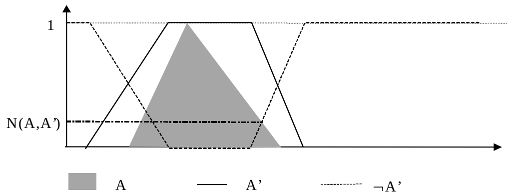


Fig. 74 : nécessité de A relative à A'

Possibilité et nécessité relatives sont deux mesures complémentaires. On peut les utiliser indépendamment pour définir un degré de compatibilité mais on peut aussi les combiner comme dans [Kelman 96]. Kelman prend en effet la moyenne de ces deux coefficients comme degré de compatibilité.

On trouve aussi des variantes de ces degrés dans [Dubois 86]. Il introduit une relation d'égalité approchée R définie par exemple par :

$$\mu_R(u_1,u_2) = \max\left(0{,}1 - f\left(\frac{d(u_1,u_2)}{\lambda}\right)\right) \text{ avec } f(0) = 0,\ f(t)<1\ \forall t<1 \text{ et } f(t) \geq 1\ \forall t \geq 1$$

Avant d'appliquer les degrés de nécessité et possibilité relatives on dilate l'ensemble flou de référence grâce à cette relation floue. On obtient alors des coefficients plus optimistes que les précédents $\Pi(A \circ R, A')$ et $N(A \circ R, A')$.

On retrouve les mêmes idées dans le degré de ressemblance introduit par Rossazza [Rossazza 90]. On a une relation d'égalité floue μ_R et on calcule deux mesures duales, la possibilité et la nécessité de vérifier cette égalité définies par :

$$\Pi(R(A,B)) = \sup_{d,d'}\left\{\min\left(\mu_R(d,d'), \pi_A(d), \pi_B(d')\right)\right\}$$

$$N(R(A,B)) = \inf_{d,d'}\left\{\max\left(\mu_R(d,d'), 1-\pi_A(d), 1-\pi_B(d')\right)\right\}$$

Enfin, lorsque l'on compare deux objets, chacun étant composé de plusieurs attributs, il faut combiner les degrés de compatibilité S_i entre attributs [Dubois 88.1]. On peut combiner ces valeurs de façon conjonctive (min) ou disjonctive (max). On peut aussi attribuer des importances relatives w_i aux différents attributs, ce qui donne pour une combinaison conjonctive $\min_i \max(1-w_i, S_i)$ et pour une combinaison disjonctive $\max_i \min(w_i, S_i)$. La compatibilité partielle des k meilleurs attributs est, elle, donnée par : $S(k) = \max_{\sigma} \min_{i=1\ldots k} S_{\sigma(i)}$, σ désignant les permutations circulaires.

Nous reviendrons sur ces notions de combinaison dans le chapitre IV.3.1 p.150.

B) Compatibilité floue

La compatibilité entre deux ensembles flous peut elle-même être vue comme un ensemble flou de [0,1] sur [0,1] [Dubois 80]. Soit A un ensemble flou de fonction d'appartenance μ_A. Pour tout singleton x, $\mu_A(x)$ peut être vue comme le degré de compatibilité de x avec A. Si maintenant on étend x à un ensemble flou A', le degré de compatibilité peut s'obtenir en appliquant le principe d'extension sur la relation précédente, ce qui donne :

$$\mu_{C(A,A')} = [\mu_A(A')](u) = \sup_{u=\mu_A(x)} \mu_{A'}(x)$$

Voici deux exemples de calcul de degrés de compatibilité flous :

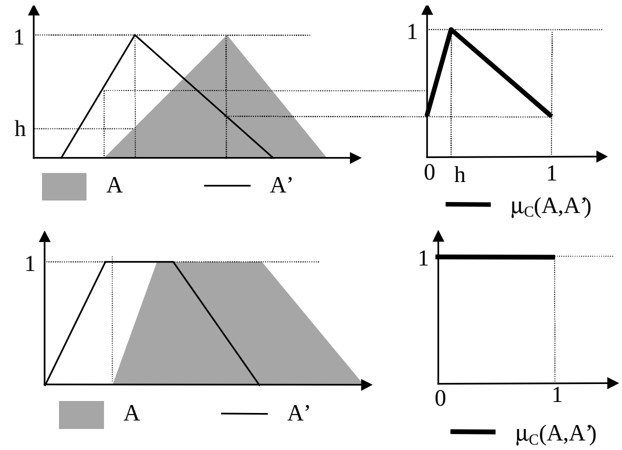


Fig. 75 : degrés de compatibilité flous

$[\mu_C(A,A')](x)$ s'interprète comme la possibilité que A' soit compatible avec A avec un degré x. Le second cas fait apparaître que dès que l'intersection entre les deux ensembles flous à comparer devient trop importante le degré de compatibilité n'est plus significatif, il vaut 1 partout. Le degré de compatibilité flou n'est pas assez nuancé, nous avons besoin pour notre application de distinguer les cas où l'intersection est normalisée.

C) Compatibilité de Yager

Yager définit les propriétés théoriques que doit vérifier selon lui une mesure de compatibilité R [Kelman 95].

- R : $X^2 \rightarrow [0,1]$
- $\forall$ $x \in X$, R(x,x)=1 (réflexivité)
- $\forall$ $(x,y) \in X^2$, R(x,y)=R(y,x) (symétrie)
- Soit d une distance, $\forall$ $(x,y,z) \in X^3$, si $d(x,z) \geq d(x,y)$ alors $R(x,z) \leq R(x,y)$
- R n'est pas forcément une fonction de d, on peut avoir $d(x,y) \geq d(z,t)$ et $R(x,y) > R(z,t)$

La propriété de symétrie est discutable dans notre cas. La contrainte réelle et la contrainte de référence ne jouent en effet pas un rôle symétrique : c'est la contrainte réelle qui doit satisfaire la contrainte de référence et pas l'inverse. La figure suivante montre un exemple où la compatibilité interprétée comme un degré de satisfaction d'une contrainte n'est pas du tout une notion symétrique.

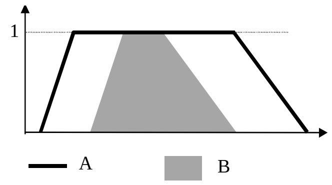


Fig. 76 : non symétrie de la notion de compatibilité

Sur ce schéma B satisfait totalement A alors que l'on ne peut pas être certain que A satisfasse B. Une définition symétrique de la compatibilité ne répond donc pas à notre besoin.

D) Compatibilité de Isomoto

Isomoto [Isomoto 97] introduit une mesure de compatibilité qui correspond plus à une notion de satisfaction. Cette mesure est en effet utilisée pour répondre à des requêtes sur une base de données d'images ou de scènes. On cherche à évaluer pour chaque objet de la base le degré de satisfaction de la requête.

Soit A l'objet de référence de la requête et A' un objet de la base à comparer à la requête. On considère les surfaces S_1 et S_2 hachurées sur le schéma suivant :

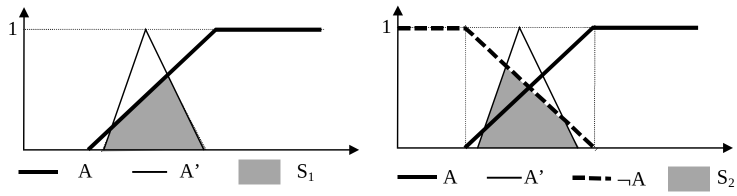


Fig. 77 : compatibilité par rapport à une requête

Le degré de compatibilité est donné par $\Pi_{A,A'} = \dfrac{S_1}{S_1 + S_2}$.

E) Degré d'inclusion

Rossazza [Rossazza 90] utilise lui un degré d'inclusion pour établir une hiérarchie de classes. L'inclusion peut être vue dans certains cas comme une compatibilité particulière, comme nous le verrons par la suite. Le degré d'inclusion d'un ensemble flou A' dans un ensemble flou A peut être défini par :

$$N_{\supseteq}(A, A') = \inf_u \left\{ \mu_{A'}(u) \Rightarrow \mu_A(u) \right\} \text{ avec une implication floue .}$$

Il y a autant de degrés d'inclusion différents que d'implications floues. Rossazza choisit l'implication de Gödel μ_G dont on rappelle la définition :

$$\mu_G(u) = 1 \text{ si } \mu_{A'}(u) \leq \mu_A(u)$$
$$\mu_G(u) = \mu_A(u) \text{ sinon}$$

Voici un exemple de calcul du degré d'inclusion par cette méthode :

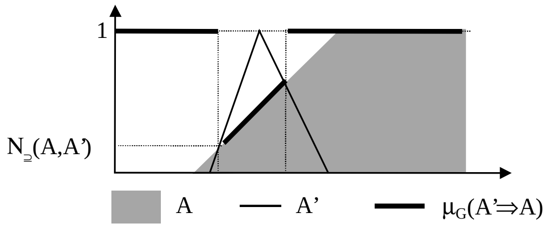


Fig. 78 : degré d'inclusion construit à partir de l'implication de Gödel

Dans [Godo 95] on utilise aussi ce type de degré d'inclusion avec une implication de Lukasiewicz I_L : $I_L(x,y)=\min(1,1-x+y)$. Le degré d'inclusion est nul dès qu'il existe une valeur x_0 telle que $\mu_{A'}(x_0)=1$ et $\mu_A(x_0)=0$.

IV.2.3 Mesures de comparaison

On pourrait trouver encore bien d'autres mesures, chacune ayant des propriétés différentes. Intuitivement, on constate que toutes ces mesures n'ont pas les mêmes significations : certaines expriment des notions proches alors que d'autres se différencient totalement. Il est donc intéressant d'avoir une vue un peu plus globale pour orienter la recherche d'une telle mesure. Dans ce souci, Rifqi [Rifqi 96] a élaboré une classification et une caractérisation des différentes mesures qu'elle qualifie de "mesures de comparaison". Nous allons nous appuyer sur ces travaux pour choisir une mesure de compatibilité adéquate.

A) Classification

Selon l'approche de Tversky [Tversky 77], pour deux ensembles A et B, leur mesure de similarité ne dépend que de leur ressemblance et de leurs différences, c'est-à-dire de $A \cap B$, A-B et B-A.

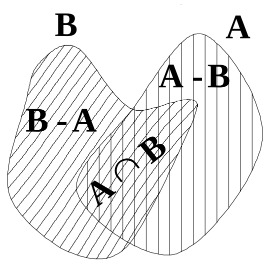


Fig. 79 : paramètres des mesures de similarité

Pour des ensembles flous, Rifqi définit les mesures de comparaison à partir d'une mesure d'ensemble flou M. Une M-mesure de comparaison est définie par :

$$S(A,B)=F(M(A\cap B),M(B-A),M(A-B))=F(u,v,w)$$

Suivant les propriétés de F, on distingue les M-mesures de similarité, de dissimilarité, de satisfiabilité, de ressemblance et d'inclusion. Ces différentes catégories s'organisent ainsi :

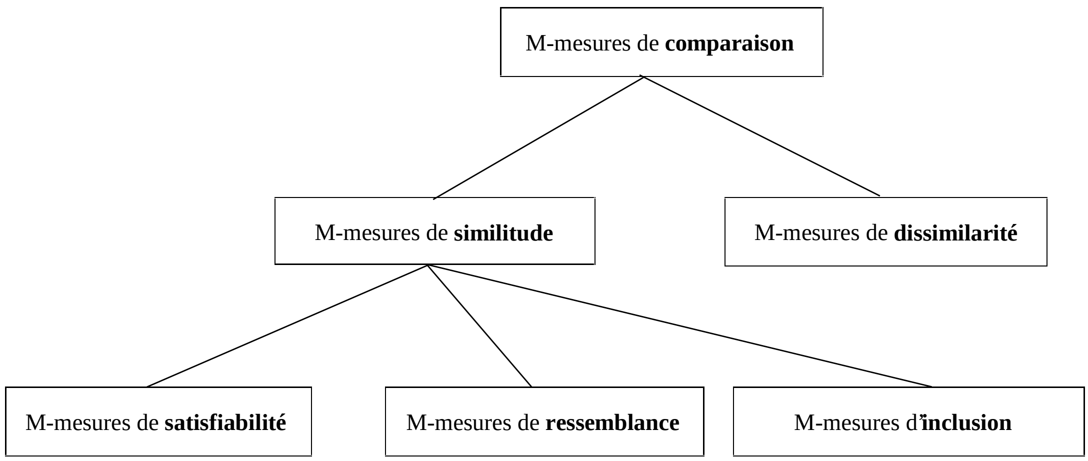


Fig. 80 : classification des M-mesures de comparaison

Les différentes M-mesures identifiées sont caractérisées par les propriétés suivantes :

- ***similitude*** : F est croissante en u, décroissante en v et w et F(0,v,w)=0.
- ***dissimilarité*** : F est indépendante de u, croissante en v et w et F(u,0,0)=0.
- ***satisfiabilité*** : mesure de similitude avec F indépendante de w et F(u,0,w)=1 si u≠0.
- ***ressemblance*** : mesure de similitude avec F(u,v,w)=F(u,w,v) et F(u,0,0)=1 si u≠0.
- ***inclusion*** : mesure de similitude avec F indépendante de v et F(u,v,0)=1 si u≠0.

B) Propriétés de la mesure de compatibilité cherchée

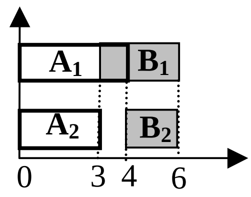
Il est donc intéressant de situer la mesure de compatibilité que l'on cherche par rapport à ce contexte. Pour cela nous allons voir les propriétés fondamentales liées aux différentes M-mesures. Nous considérons que A correspond à la contrainte du modèle à satisfaire et B à la contrainte observée. S(A,B) est la mesure de compatibilité de B par rapport à A.

Tout d'abord, la notion de satisfaction de contrainte n'est pas symétrique. Ce n'est en effet pas la même chose de dire que A satisfait B ou que B satisfait A. Si B est incluse dans A par exemple, alors B satisfait totalement A mais la réciproque n'est pas vraie (cf. Fig. 76 p.138). On doit donc avoir F(u,v,w)≠F(u,w,v). Ceci exclut les mesures de ressemblance.

D'autre part, S dépend bien de l'intersection de A et B. Considérons les deux cas suivants sur des intervalles classiques :

1. $A_1=[0,4]$ et $B_1=[3,6] \Rightarrow u=[3,4]$, $v=[4,6]$ et $w=[0,3]$

2. $A_2=[0,3]$ et $B_2=[4,6] \Rightarrow u=\emptyset$, $v=[4,6]$ et $w=[0,3]$



Dans le cas 1, B_1 peut satisfaire A_1 alors que dans le cas 2 c'est impossible puisque aucune valeur de B_2 n'est compatible avec A_2. Le seul paramètre qui diffère entre les deux cas est l'intersection, et pourtant le résultat de la comparaison diffère, c'est donc que la mesure de compatibilité dépend de la mesure de l'intersection u. Une mesure de dissimilarité ne peut pas convenir pour exprimer cette notion de satisfaction. On peut ajouter que S est croissante avec la mesure de l'intersection. Plus il y a de valeurs réelles qui vérifient la contrainte, plus la contrainte a de chances d'être satisfaite.

Enfin, nous allons voir que S dépend du paramètre v. v représente en effet toutes les valeurs réelles possibles (d'après la connaissance que l'on a de la réalité) mais qui ne satisfont pas la contrainte A. Ce sont toutes les valeurs qui ne conviennent pas. Plus ces valeurs sont nombreuses, moins on a de chance de vérifier la contrainte. Ceci apparaît sur le schéma suivant où les paramètres u et w sont constants entre B et B'. Seul v varie et intuitivement le résultat de S doit lui aussi être modifié pour exprimer une satisfaction.

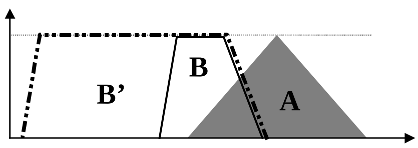


Fig. 81 : S(A,B) dépendant de v

B a beaucoup plus de chances de satisfaire A que B', S ne correspond donc pas à une mesure d'inclusion qui doit être indépendante de v. On a même $M(B-A)<M(B'-A)$ et $S(A,B)>S(A,B')$, F est décroissante en v.

La classe de mesures la plus spécifique qui nous reste est donc la satisfiabilité. Nous allons voir si les propriétés des mesures de satisfiabilité conviennent pour exprimer la compatibilité par rapport à une contrainte de référence. Pour cela S doit vérifier les propriétés suivantes :

1. $0 \leq S(A,B) \leq 1$
2. S croissante avec $u=M(A \cap B)$
3. S décroissante en $v=M(B-A)$
4. S indépendante de $w=M(A-B)$
5. $S(A,B)=0$ si $A \cap B=\varnothing$
6. $S(A,B)=1$ si $B-A=\varnothing$ et $A \cap B \neq \varnothing$

Le degré de compatibilité est positif et inférieur au degré de satisfaction global lui-même inférieur à 1, donc la propriété 1 est vérifiée. On a déjà montré que les propriétés 2 et 3 doivent être vérifiées par notre mesure de satisfaction S. Nous allons voir maintenant la propriété 4, l'indépendance de S par rapport à w. w correspond en effet à toutes les valeurs qui satisfont la contrainte de référence mais qui de toutes façons sont exclues des valeurs réelles possibles. Si B est précisé par la suite par de nouvelles informations, l'ensemble ne peut que se restreindre en un ensemble $B' \subseteq B$ et on aura $A-B \subseteq A-B'$ donc aura $M(A-B) \leq M(A-B')$. Les valeurs exclues de la solution au départ restent exclues par la suite, elles n'influent pas sur la satisfaction de la contrainte.

Ceci est illustré par le schéma suivant qui montre le cas particulier d'une observation B précise. On connaît alors le degré de satisfaction de A, c'est aussi le degré de compatibilité de B par rapport à A. Ce degré reste constant alors que $M(A-B) \neq M(A'-B)$.

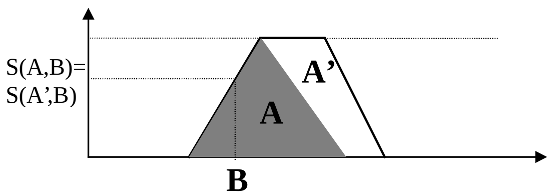


Fig. 82 : S(A,B) indépendant de w

La propriété 5 est évidente. Si l'intersection de A et B est nulle, il n'y a aucune des valeurs observées qui vérifie la contrainte, la satisfaction est donc nulle. Pour la propriété 6, la condition $A \cap B \neq \varnothing$ et $B-A = \varnothing$ revient à dire qu'il existe des valeurs de B qui satisfont A mais aucune ne viole la contrainte A. A ne peut donc qu'être satisfaite, S vaut bien 1.

La mesure que l'on cherche correspond donc bien à la classe des M-mesures de satisfiabilité. On peut construire de nombreuses mesures de satisfiabilité. Pour cela il faut d'abord choisir une mesure floue M et un opérateur de différence ensembliste floue. Nous allons voir les différents choix possibles.

C) Mesures floues

Une mesure floue est une fonction M définie sur les ensembles flous d'univers numérique, qui prend ses valeurs sur $\Re^+$ et telle que $M(\emptyset) = 0$ et si $B \subseteq A$ alors $M(B) \leq M(A)$. Les principales mesures floues vérifient en plus $M(A) = 0 \Leftrightarrow A = \emptyset$. En voici une liste non exhaustive [Rifqi 96] :

$$M_1(A) = \int \mu_A(x)dx$$

$$M_2(A) = \sup_x \mu_A(x)$$

$M_3(x) = |\mathrm{supp}(A)|$ où | . | est le cardinal et supp le support.

D) Différences ensemblistes floues

La différence d'ensembles non flous peut se définir de deux manières équivalentes : $A\text{-}B = A \cap (\neg B)$ ou $A\text{-}B = \bigcap \{C, A \subseteq (B \cup C)\}$. Ces deux définitions étendues aux ensembles flous ne donnent pas les mêmes opérateurs de différence. On retiendra deux de ces opérateurs de différence en ajoutant comme condition la continuité. Ces opérateurs sont :

$\mu_{1(A-B)} = \max(0, \mu_A - \mu_B)$ (différence de Zadeh)

$$\mu_{2(A-B)} = \min(\mu_A, 1 - \mu_B)$$

IV.2.4 Mesures de satisfiabilité proposées

A) Mesures de satisfiabilité existantes

Nous allons maintenant essayer de choisir une ou plusieurs mesures de satisfiabilité. Parmi toutes les mesures de compatibilité présentées en IV.2.2 p.135, une seule correspond en fait à la définition d'une mesure de satisfiabilité : il s'agit de la mesure de Isomoto, que nous examinerons de plus près par la suite. Rifqi [Rifqi 96] a recensé les principales mesures de satisfiabilité existantes. Nous allons regarder si celles-ci peuvent convenir.

La plus simple est donnée pour une mesure floue M correspondant à la hauteur par :

1. $S(A,B) = 1 - \sup_{x/\mu_A(x)=0} \mu_B(x) = 1 - M(B - A)$

avec la différence floue : $\mu_{B-A}(x) = \mu_B(x)$ *si* $\mu_A(x) = 0$ et $\mu_{B-A}(x) = 0$ sinon.

L'inconvénient de cette mesure de satisfiabilité est qu'elle peut être nulle alors que l'intersection de A et de B est non vide comme sur le schéma suivant :

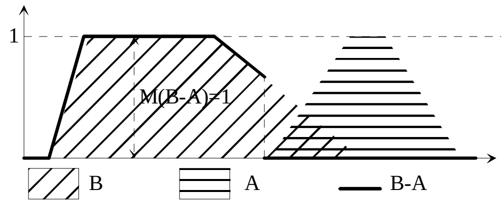


Fig. 83 : satisfiabilité nulle

L'observation B peut satisfaire la contrainte A puisque l'intersection est non vide mais pourtant la satisfiabilité est nulle. Cette mesure ne convient donc pas dans notre cas.

On retrouve le même phénomène avec une mesure de satisfiabilité construite sur le même principe S(A,B)=1-M(B-A) mais en utilisant la différence de Zadeh. Il est cependant préférable d'utiliser cette différence de Zadeh qui, contrairement à la précédente, est continue.

Ce sont les seules M-mesures de satisfiabilité proposées pour la mesure floue M correspondant à la hauteur de l'ensemble flou. Nous allons maintenant voir celles qui sont proposées avec la mesure floue qui est la surface de l'ensemble. Une première mesure est :

2. $S(A,B) = \dfrac{M(A \cap B)}{M(A \cap B) + M(B - A)} = \dfrac{M(A \cap B)}{M(B)}$ avec la différence de Zadeh.

Le problème rencontré avec les mesures précédentes est alors résolu mais il en apparaît un nouveau. Dans notre application, les observations pourront être très imprécises et on peut avoir une information sur un événement du type : "l'événement a lieu avant t_0". Dans ce cas, l'intervalle flou B est non borné à gauche et donc sa surface est infinie. La satisfiabilité sera alors nulle quelle que soit la valeur de l'intersection de A et B, ce qui n'est pas satisfaisant. Même si toutes les valeurs de A sont encore possibles pour B, A est dite non satisfiable par B. Ceci est illustré sur le schéma suivant :

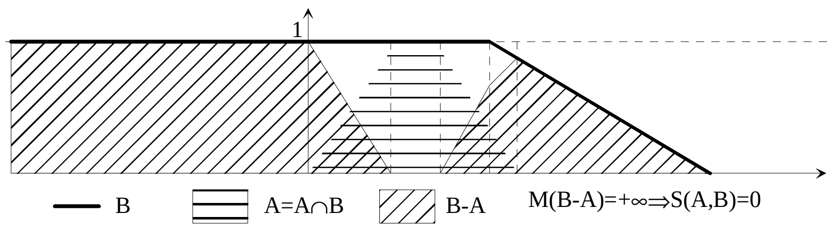


Fig. 84 : problème des ensembles non bornés

La mesure de Isomoto est une mesure de satisfiabilité construite de la même façon par :

3. $S(A,B) = \dfrac{M(A \cap B)}{M(A \cap B) + M(B - A)}$ mais avec la différence ensembliste définie par :

$\mu_{B-A}(x) = \min(\mu_B, 1-\mu_A)$.

On a aussi S(A,B)=0 dès que M(B)=+∞ même si $A \cap B \neq \emptyset$.

Rifqi introduit aussi une nouvelle mesure de satisfiabilité ayant les mêmes propriétés que les deux précédentes et donc les mêmes inconvénients mais qui décroît un peu moins rapidement avec M(B-A). Cette mesure vaut :

$$4.\ S(A,B) = \frac{2}{\pi}\arctan\left(\frac{M(A \cap B)}{M(B-A)}\right).$$

Aucune des mesures citées ne convient parfaitement à notre problème. En revanche elles font apparaître de nouvelles propriétés souhaitables pour notre mesure de satisfiabilité. Nous allons donc résumer toutes les caractéristiques que devrait respecter notre mesure.

B) Propriétés à satisfaire

S est une mesure de satisfiabilité dont nous rappelons les propriétés indispensables :

1. $0 \leq S(A,B) \leq 1$
2. S croissante en $A \cap B$
3. S décroissante en B-A
4. S indépendante de A-B
5. $A \cap B = \emptyset \Rightarrow S(A,B)=0$
6. $(B \subseteq A,\ B \neq \emptyset$ et $A \neq \emptyset) \Rightarrow S(A,B)=1$

A ces propriétés nous en ajoutons une qui est apparue nécessaire avec les remarques suscitées par les mesures de satisfiabilité précédentes. Dès que l'on a une observation B qui n'est pas contradictoire avec la contrainte A, la satisfiabilité doit être non nulle. Même si B est très imprécis, s'il est un tant soit peu compatible avec A, S est non nulle. Ceci se traduit par :

5'. $S(A,B)=0 \Rightarrow A \cap B = \emptyset$

Enfin, les mesures de satisfiabilité sont définies pour des ensembles flous normalisés, ce qui n'est pas notre cas. Suite à l'application du mécanisme de propagation, les contraintes peuvent être non normalisées, ce qui nous donne une borne supérieure de la satisfaction globale des contraintes. La satisfaction d'une contrainte particulière ne peut pas être supérieure à cette satisfaction globale. La satisfiabilité sera donc bornée par ce degré de satisfaction global qui correspond à la hauteur H(A) de la contrainte A. La propriété 1 devient :

1'. $0 \leq S(A,B) \leq H(A)$

La propriété 6 est aussi modifiée en conséquence. On la renforce en lui ajoutant aussi sa réciproque. La satisfiabilité n'est maximale que si la contrainte est certaine d'être satisfaite, c'est-à-dire si (B⊆A et A∩B≠∅). La propriété 6 devient :

6'. (B⊆A et A∩B≠∅)⇔S(A,B)=H(A)

C) Nouvelles mesures de satisfiabilité

Toutes ces propriétés étant fixées, nous proposons quelques mesures de satisfiabilité inspirées de celles déjà existantes. Voici la liste de ces mesures : pour chacune on peut utiliser les différences μ_1 ou μ_2 définies précédemment :

1. $S_1(A,B) = \frac{1}{2}\left[M(A \cap B) + H(A) - M(B - A)\right]$ avec M=hauteur.

2. $S_2(A,B) = H(A)\dfrac{\arctan(M(A \cap B))}{\arctan(M(A \cap B)) + \arctan(M(B - A))}$ avec M=surface.

3. $S_3(A,B) = H(A)\dfrac{M(A \cap B)}{M(A \cap B) + \min[M(B - A), K]}$ avec M=surface et une constante K représentant le seuil à partir duquel on considère qu'une surface est infinie.

La première mesure est construite à partir de la mesure de satisfiabilité classique 1 utilisant la hauteur comme mesure floue : S(A,B)=1-M(A∩B). Comme on l'a vu en IV.2.4A) p. 143, cette mesure ne nous convient pas car elle ne vérifie pas la propriété 5'. Nous souhaitons cependant avoir une mesure simple comme celle-ci, linéaire et utilisant la hauteur comme mesure floue. Il faut donc introduire la mesure de l'intersection pour vérifier la propriété 5'. Notre mesure est croissante avec l'intersection donc nous lui donnons un coefficient positif. La mesure précédente était proposée pour des ensembles normalisés, ce qui n'est pas notre cas donc nous remplaçons l'élément 1 par la hauteur de l'ensemble de référence H(A). Enfin il reste à vérifier la propriété 1'. Pour cela on multiplie l'expression obtenue par 1/2.

Les deuxième et troisième mesures sont obtenues à partir des mesures 2 et 3 qui diffèrent uniquement par la mesure de différence utilisée. Pour ces deux mesures on a le même problème, dès que M(B-A) est infinie la satisfiabilité est nulle, même si l'intersection est non nulle. On propose deux solutions : soit on ajoute une fonction pour avoir une valeur finie même dans le cas d'un ensemble infini, soit on borne la mesure par une constante. La première solution correspond à notre deuxième mesure de satisfiabilité où l'on a utilisé la fonction arctan. La deuxième solution correspond à notre mesure 3.

Il faut encore multiplier ces mesures par le coefficient H(A) pour vérifier la propriété 1' portant sur le domaine de l'image de S(A,B).

Voici, dans tous les cas de figures possibles, le comportement de ces différentes mesures. Un premier schéma donne les mesures de satisfiabilité S_i en prenant la différence ensembliste de Zadeh.

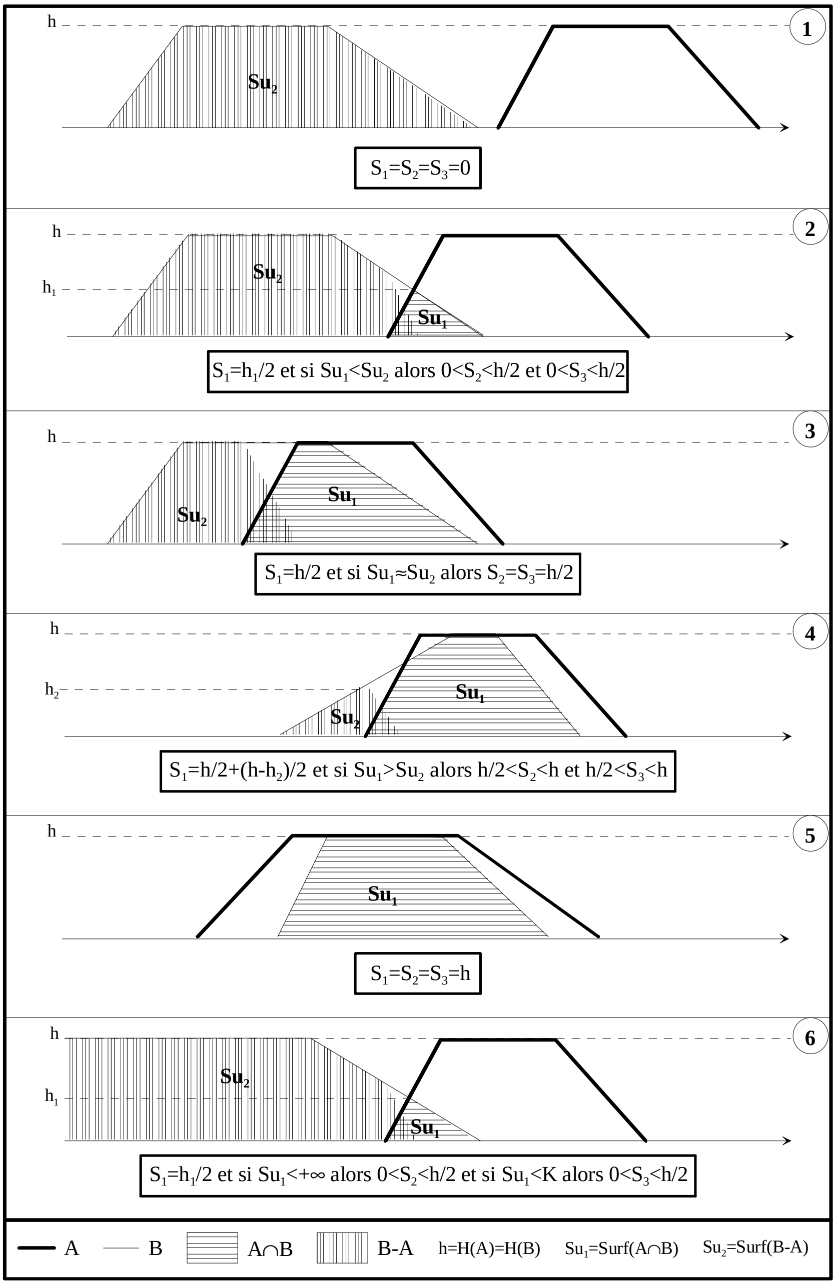


Fig. 85 : comportement des différentes mesures de satisfiabilité

Si on prend l'autre définition pour la différence ensembliste, seul le cas ⑤ de la figure Fig. 85 est modifié. Il se décompose en fait en deux cas, suivant que B est inclus dans le noyau de A ou non. Voici ce que cela donne :

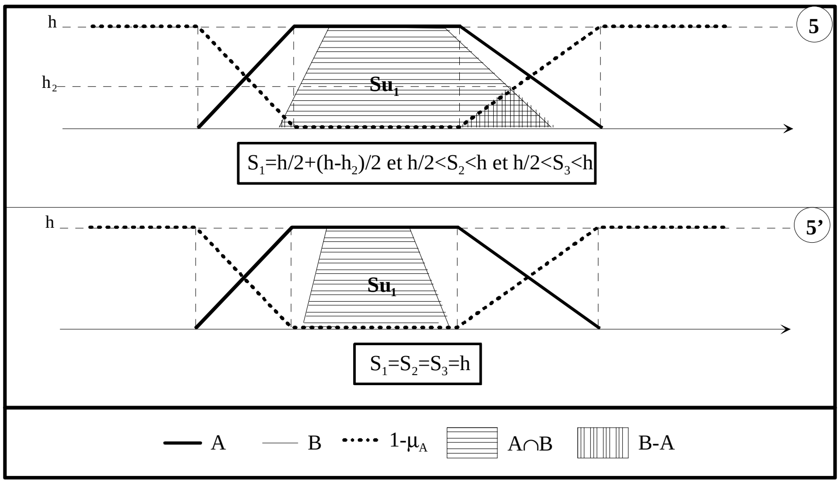


Fig. 86 : comportement des différentes mesures de satisfiabilité avec μ_2

Le choix d'une mesure de satisfiabilité plutôt qu'une autre ne pourra se faire qu'après avoir testé ces différentes mesures. On peut déjà donner les principales différences de comportement. Le choix de la mesure floue surface apporte plus de nuances sur le résultat. Dans le cas ③ par exemple, si on prend la hauteur comme mesure floue on peut diminuer autant que l'on veut la borne inférieure de B sans pour autant modifier le résultat puisque les hauteurs ne sont pas modifiées. En revanche, si l'on prend la surface comme mesure floue, la satisfiabilité diminue avec la borne inférieure de B, ce qui est logique puisqu'il y a plus de valeurs de B qui ne satisfont pas A.

Entre S_2 et S_3, les différences sont minimes. S_2 est plus souple mais risque d'augmenter les temps de calculs. D'autre part, avec S_2 on n'a pas besoin de déterminer de constante K. Pour S_3 il faudra trouver la valeur de K qui correspond en fait à la surface du plus grand ensemble flou n'ayant aucune borne infinie.

Enfin, le choix de μ_2 introduit une nuance pour le cas où B est inclus dans A. Dans ce cas en effet, B est certain de satisfaire A avec le degré h uniquement si B est inclus dans le noyau de A, ce qui est en faveur de μ_2.

On préconise donc l'utilisation de S_2 avec μ_2, ce qui donnera le plus de précision et ne nécessite pas la définition d'une constante. Si l'on veut une évaluation plus rapide on choisira S_1, toujours avec la différence μ_2. Ce choix dépend fortement du contexte de l'application, de la performance attendue en matière de temps de calcul et de précision. Ces deux paramètres sont souvent antagonistes.

IV.3 Mesures d'agrégation

On sait maintenant évaluer la satisfiabilité par rapport à chaque contrainte, il reste à choisir un opérateur de combinaison pour calculer la compatibilité partielle d'un nœud de gabarit. L'opérateur d'agrégation des contraintes au niveau d'un nœud est unique et sera choisi par la suite parmi les différents opérateurs présentés ici. Le degré de compatibilité partiel d'un nœud est appelé évaluation "marginale". L'agrégation constitue la deuxième étape dans l'évaluation des gabarits. Il s'agit, à partir des évaluations marginales, de déduire une évaluation globale du gabarit.

Cette évaluation globale est obtenue par une agrégation multicritère. Un critère correspond à un nœud donc à une activité du gabarit qu'il faut reconnaître. La reconnaissance globale du gabarit dépend de la reconnaissance de chacun de ses nœuds, c'est elle qu'on évalue. Il s'agit de l'approche "cardinale" et non de l'approche "ordinale" de l'évaluation multicritère [Kelman 96]. On ne cherche en effet pas à comparer plusieurs situations vis-à-vis d'un gabarit mais bien à "noter" l'hypothèse du gabarit dans une situation donnée. Les évaluations marginales étant des nombres réels, on ne s'intéresse qu'à des opérateurs d'agrégation numériques.

Chaque décomposition d'une activité en sous-activités élémentaires peut avoir une signification particulière. Prenons l'exemple du gabarit suivant :

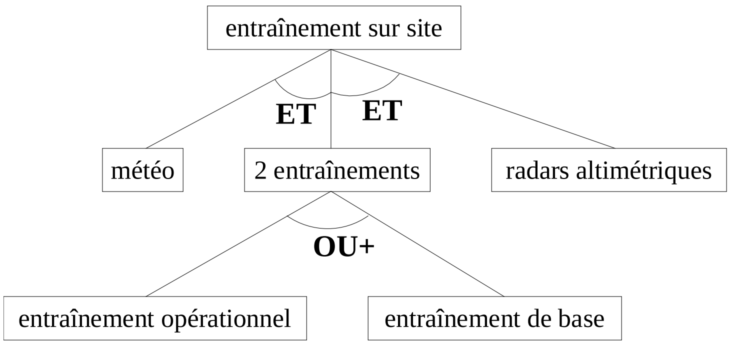


Fig. 87 : exemple du gabarit entraînement sur site

Dans cet exemple, 2 agrégations de nature différentes apparaissent : ***ET*** et ***OU+***. L'agrégation ***ET*** signifie que pour reconnaître une activité ***entraînement sur site*** il faudra reconnaître les 3 activités qui la composent : ***reconnaissance météo***, ***2 entraînements*** et ***radars altimétriques***. On peut nuancer en disant que l'activité ***2 entraînements*** est très caractéristique de l'activité ***entraînement sur site*** et qu'à elle seule elle permet quasiment de reconnaître le gabarit. La deuxième agrégation ***OU+*** signifie que l'une des deux activités ***entraînement opérationnel*** ou ***entraînement de base*** suffit pour reconnaître l'activité globale ***2 entraînements***, mais si l'on reconnaît les deux sous-activités alors l'hypothèse de l'activité ***2 entraînements*** doit être renforcée.

Cet exemple montre qu'il n'existe pas une fonction d'agrégation universelle. Il faut déterminer chaque agrégation en fonction de la signification que l'on souhaite lui accorder en terme de reconnaissance. L'agrégation doit être autant que possible le reflet du raisonnement humain. Le type d'agrégation à choisir ne peut donc être obtenu qu'en interrogeant l'expert. Mais pour cela il faut connaître les différents opérateurs d'agrégation possibles ainsi que leurs propriétés caractéristiques. Dans un premier temps, nous avons recensé les différents opérateurs utilisables.

IV.3.1 Opérateurs d'agrégation

La définition la plus générale d'un opérateur d'agrégation est : un opérateur d'agrégation est une fonction de X^n dans X où X est un univers métrique muni d'un ordre. On restreint légèrement cette définition en prenant des fonctions continues non décroissantes et avec X=[0,1] puisque les évaluations marginales sont dans [0,1]. La majorité des opérateurs d'agrégation vérifie ces contraintes.

Dans la littérature, il existe de multiples opérateurs d'agrégation [Bouchon 98], [Kelman 96], [Yager 91] et [Dubois 88.2]. L'objectif ici est de caractériser de façon simple l'attitude de chacun des opérateurs. Ceci nous permettra de nous limiter pour les gabarits aux opérateurs les mieux adaptés.

Il existe un ordre partiel sur les opérateurs d'agrégation [Bloch 96.1]. Certains opérateurs sont en effet ordonnés, ils forment une échelle de référence, les autres sont non ordonnés et varient le long de cette échelle. Les premiers sont dits ***"à attitude fixe"*** ou ***"constante"***. On trouve dans l'ordre croissant : les conjonctions, les opérateurs de compromis et les disjonctions. Les seconds sont dits ***"à attitude variable"***. Leur attitude varie en fonction des valeurs à agréger : ils peuvent varier d'une attitude conjonctive à disjonctive.

Nous présentons ici les différents opérateurs d'agrégation suivant la classification opérateurs ***à attitude fixe*** ou ***variable*** et suivant leur ordre pour la première catégorie. Pour chaque opérateur nous donnons sa définition dans le cas de l'agrégation de deux critères puis son extension à des critères pondérés. Pour les opérateurs d'agrégation $h^{(2)}$ associatifs, leur extension à n critères $h^{(n)}$ s'obtient par récurrence :

$$h^{(n)}(a_1,\ldots,a_n) = h^{(2)}(h^{(n-1)}(a_1,\ldots,a_{n-1}),a_n)$$

Quand elle existe, nous ne préciserons l'extension à n critères que pour les opérateurs non associatifs.

Notations : C_i est le $i^{ème}$ critère, a_i est l'évaluation marginale du critère C_i et w_i est le poids du critère C_i.

A) Opérateurs à attitude constante

a) Opérateurs conjonctifs

Les opérateurs conjonctifs [Yager 91], [Fodor 94], correspondent à l'opérateur logique "et". Ce sont des t-normes. Une t-norme T est un opérateur commutatif, associatif, monotone et tel que T(a,1)=a pour tout a [Bouchon 94]. De plus toute t-norme vérifie $T_{weber} \leq T(x,y) \leq \min(x,y)$. Les t-normes les plus usuelles sont [Dubois 94] :

1. $T(a_1,a_2)=\min(a_1,a_2)$ (minimum)
2. $T_{weber}(x,1)=x$, $T_{weber}(1,y)=y$ et $T_{weber}(x,y)=0$ si $x \neq 1$ et $y \neq 1$
3. $T(a_1,a_2)=a_1.a_2$ (produit)
4. $T(a_1,a_2)=\max(0,a_1+a_2-1)$ (conjonction linéaire)

Il existe aussi une famille de t-normes paramétrées, les t-normes de Frank [Franck 79], [Chauvin 95]. On définit une t-norme de Frank pour tout $r \in]0,1[\cup]1,+\infty[$ par :

5. $$T_r(a_1,\ldots,a_n) = \log_r\left[1+\frac{\prod_i \left(r^{a_i}-1\right)}{(r-1)^{n-1}}\right]$$

T_r est décroissante avec le coefficient r. En faisant varier r on obtient les t-normes 1,3 et 4 données précédemment ($r \to 0$, $r \to 1$ et $r \to +\infty$).

Si les critères sont pondérés, on utilise généralement l'opérateur min pondéré de Dubois [Dubois 88.2] : $T(a_1,\ldots,a_n)=\min_i\left[\max[(1-w_i),a_i]\right]$ avec $\max_i(w_i)=1$. Ceci revient à appliquer l'opérateur min sur des évaluations marginales modifiées par leur poids.

b) Opérateurs disjonctifs

Ces opérateurs correspondent au "ou" logique [Yager 91], [Fodor 94]. Ils sont exprimés par des t-conormes $\perp$, opérateurs associatifs, commutatifs, monotones et tels que $\perp(a,0)=a$ pour tout a [Bouchon 94]. A toute t-norme donnée T on peut associer une t-conorme $\perp$ dite duale de T : $\perp(a_1,a_2)=1-T(1-a_1,1-a_2)$.

On obtient ainsi les agrégations disjonctives duales des conjonctions précédentes :

1. $\perp(a_1,a_2)=\max(a_1,a_2)$ (maximum)

2. $\perp_{weber}(x,0)=x$, $\perp_{weber}(0,y)=y$ et $\perp_{weber}(x,y)=1$ si $x\neq 0$ et $y\neq 0$

3. $\perp(a_1,a_2)=a_1+a_2-a_1.a_2$ (somme probabiliste)

4. $\perp(a_1,a_2)=\min(1,a_1+a_2)$ (somme bornée)

5. $\perp_r(a_1,\ldots,a_n)=1-\log_r\left[1+\frac{\prod_i\left(r^{(1-a_r)}-1\right)}{(r-1)^{n-1}}\right]$

Toute t-conorme vérifie la propiété $\max \leq \perp \leq \perp_{Weber}$. Si les critères sont pondérés, on utilise généralement la disjonction max pondéré de Dubois [Dubois 88.2] : $\perp(a_1,\ldots,a_n)=\max_i\left[\min[w_i,a_i]\right]$ avec $\max_i(w_i)=1$. C'est l'opération duale du min pondéré.

c) Opérateurs de compromis

Les opérateurs de disjonction et de conjonction sont utilisés lorsque l'on adopte une attitude bien tranchée. Pour une disjonction, il suffit de vérifier un des critères et pour une conjonction il faut tous les vérifier. Pour toutes les positions intermédiaires on choisit des opérateurs de compromis entre une conjonction pure et une disjonction pure. Il existe plusieurs types d'opérateurs de compromis.

Par définition un opérateur de compromis m est un opérateur d'agrégation commutatif, non décroissant en chacun de ses arguments et qui vérifie min<m<max. L'effet principal des opérateurs de compromis est un effet compensatoire entre les évaluations marginales faibles et fortes, la tendance est de ramener les évaluations marginales vers le centre.

i) Moyennes classiques

Les opérateurs de moyenne classiques sont obtenus en ajoutant à la définition précédente la croissance stricte, la continuité et la bisymétrie [Dubois 88.2]. Une fonction m est bisymétrique ssi :

$$\forall(x,y,z,t),\ m[m(x,y),m(z,t)]=m[m(x,z),m(y,t)]$$

Tout opérateur de moyenne ainsi défini peut s'écrire :

$m(a_1,\ldots,a_n) = k^{-1}\left(\sum_{i=1}^{n} k(a_i)\right)$ où k est une fonction continue strictement monotone.

Généralement on prend $k(x)=x^{\alpha}$. On obtient alors la famille des moyennes quasi-arithmétiques dont les plus connues sont :

1. la moyenne harmonique pour α=-1
2. la moyenne géométrique pour $\alpha \to 0$
3. la moyenne arithmétique pour α=1
4. la moyenne quadratique pour α=2

Toutes ces moyennes sont ordonnées suivant les α croissants.

On peut ajouter des poids aux critères, l'opérateur devient :

$m(a_1,\ldots,a_n) = k^{-1}\left(\sum_{i=1}^{n} w_i k(a_i)\right)$ avec k toujours continue et strictement monotone et $\sum_{i=1}^{n} w_i = 1$.

ii) Médianes

Les médianes sont définies dans le cas général par : $med(a_1,\ldots,a_{2n+1})=a_{(n)}$ où $a_{(1)} \leq a_{(2)} \leq \ldots \leq a_{(2n+1)}$ sont les a_i ordonnés.

Pour 2 paramètres, elles sont paramétrées et s'écrivent : $med(a_1,a_2,\alpha)=a_{(2)}$, avec $\alpha \in]0,1[$. Dans le cas de deux paramètres, les médianes sont les seuls opérateurs de compromis associatifs [Dubois 88.2].

Pour n pair et n>2, la médiane n'est plus paramétrée et vaut : $med(a_1,\ldots,a_n)= \frac{1}{2}(a_{(n/2)}+a_{(n/2+1)})$

L'intérêt des médianes par rapport aux moyennes classiques est que l'on ne risque pas de fournir un résultat très éloigné de chacune des évaluations marginales.

iii) OWA

Les opérateurs OWA, **O**rdered **W**eighted **A**veraging **O**perators, introduits par Yager [Yager 88], sont des opérateurs de compromis idempotents. Ils sont paramétrés par un vecteur de poids (p_i) vérifiant $\Sigma p_i=1$. L'opérateur OWA se calcule de la manière suivante :

$OWA(a_1,\ldots,a_n)= \Sigma p_i b_i$ où b_i est le $i^{ème}$ des a_j en ordonnant les a_j dans l'ordre décroissant.

On retrouve les opérateurs min, max, moyenne arithmétique et médiane en choisissant des vecteurs de poids particuliers. On peut caractériser un opérateur OWA par une mesure d'optimisme, indiquant si l'agrégation obtenue est plutôt pessimiste (proche du min) ou optimiste (proche du max). Cette mesure d'optimisme vaut :

$Opt(p)= \Sigma p_i h_n(j)$ avec $h_n(j)=(n-j)/(n-1)$

On peut adopter la démarche inverse et construire un opérateur OWA à partir d'une mesure d'optimisme donnée. Il existe une méthode de calcul pour déterminer le vecteur de poids à partir de la mesure d'optimisme (cf. [O'Hagan 90] pour plus de détails).

Le principal intérêt des opérateurs OWA est qu'ils permettent de réaliser des agrégations partielles auxquelles on peut attacher une sémantique [Yager 95]. On peut par exemple exprimer que l'on souhaite prendre en compte au moins la moitié des critères. Un quantificateur flou est un ensemble flou de [0,1] sur [0,1] qui permet d'exprimer des expressions linguistiques comme "la plupart", "quelques uns"… On peut construire un opérateur OWA à partir d'un quantificateur linguistique Q. Cet OWA est l'agrégation des Q meilleurs critères. Le vecteur de poids s'obtient à partir de Q par : $p_i=\mu_Q(i/n)-\mu_Q((i-1)/n)$.

Enfin on peut étendre les OWA à des opérateurs d'agrégation pondérés [Kelman 96]. Ils deviennent alors non commutatifs mais permettent d'introduire les importances relatives des différents critères. On peut alors exprimer que Q des critères importants doivent être vérifiés. On associe aux évaluations marginales (a_i) le vecteur des importances relatives (v_i) avec $\max(v_i)=1$. On ordonne ensuite les (a_i) dans l'ordre décroissant pour obtenir le vecteur (b_i). On réordonne aussi les importances relatives en un vecteur (u_i) où u_j est l'importance qui correspond à l'évaluation marginale $b_j=a_k$, donc $u_j=v_k$. Le vecteur de pondération de l'opérateur OWA s'obtient alors par :

$$p_j(x) = \mu_Q\left(\frac{\sum_{k=1}^{j} u_k}{T}\right) - \mu_Q\left(\frac{\sum_{k=1}^{j-1} u_k}{T}\right) \text{ avec } T = \sum_{k=1}^{n} u_k$$

d) Intégrales floues

On peut construire des opérateurs d'agrégation multicritère à partir d'intégrales floues. Pour définir une intégrale floue, il faut d'abord choisir une mesure floue g telle que $g(c_i)$ représente le poids du critère c_i et $g(\{c_i,c_j,c_k\})$ représente le poids d'un groupe de critères [Grabisch 95.1]. Le poids d'un groupe de critères permet d'exprimer des interactions, des complémentarités ou des incompatibilités entre critères.

Les intégrales floues ne sont pas des opérateurs de compromis au sens strict de la définition puisqu'ils ne vérifient pas forcément la commutativité (comme tout opérateur pondéré). Mais ils ont une attitude semblable aux opérateurs de compromis puisqu'ils sont toujours compris entre les opérateurs minimum et maximum. On distingue deux types d'intégrales floues, les intégrales de Sugeno et les intégrales de Choquet.

i) Intégrales de Sugeno

Appliquée à de l'agrégation multicritère, l'intégrale floue de Sugeno construite avec la mesure floue g est :

$$F(a_1,\ldots,a_n) = \max_{i=1\ldots n}\left[\min\left(a_{(i)}, g(\Gamma_{(i)})\right)\right] \text{ où les } a_{(1)} \leq \ldots \leq a_{(n)} \text{ sont les } a_i \text{ ordonnés,}$$

$$\Gamma_{(i)} = \left\{c_{(i)},\ldots,c_{(n)}\right\} \text{ avec } c_{(i)} \text{ est le critère associé à l'évaluation } a_{(i)}$$

Les intégrales floues de Sugeno sont des opérateurs continus, monotones, idempotents, ils vérifient les conditions aux limites (F(0,…,0)=0 et F(1,…,1)=1) et ils vérifient la propriété de chaînage ordonné avec permutation [Grabish 93].

ii) Intégrales de Choquet

Appliquée à de l'agrégation multicritère, l'intégrale floue de Choquet construite avec la mesure floue g est :

$$F(a_1,\ldots,a_n) = \sum_{i=1}^{n} a_{(i)}\left(g(\Gamma_{(i)}) - g(\Gamma_{(i+1)})\right) \text{avec } g(\Gamma_{(1)}) = 1 \text{ et } g(\Gamma_{(n+1)}) = 0$$

Les intégrales de Choquet vérifient les mêmes propriétés que les intégrales de Sugeno avec en plus la stabilité par transformation linéaire positive :
$\forall r > 0$, $F(ra_1+t,\ldots,ra_n+t)=rF(a_1,\ldots,a_n)+t$, c'est-à-dire que le résultat de l'agrégation n'est pas affecté par un changement d'échelle [Grabish 94].

Il existe une caractérisation des intégrales de Choquet : ce sont les transformations linéaires positives de mêmes pentes et à zéros comonotones [Grabish 94]. De plus on peut démontrer qu'il y a équivalence entre les opérateurs OWA et les intégrales de Choquet commutatives [Grabish 95.2].

e) Opérateurs de compensation

Les opérateurs de compensation sont une combinaison d'une conjonction I et d'une disjonction U [Chauvin 95]. I et U doivent être pondérés si on veut un opérateur de compensation pondéré. La forme générale d'un opérateur de compensation est : $I^{\gamma}U^{1-\gamma}$ avec $\gamma \in [0,1]$. Le comportement de cet opérateur varie suivant la valeur de γ et en fonction des évaluations marginales. On calcule les deux valeurs seuils pour la valeur de γ :

$$\gamma_1 = \frac{\ln(\min) - \ln(U)}{\ln(I) - \ln(U)} \text{ et } \gamma_2 = \frac{\ln(\max) - \ln(U)}{\ln(I) - \ln(U)}$$

γ_1 et γ_2 dépendent des évaluations marginales. L'attitude de l'opérateur de compensation dépend de la situation de γ par rapport aux deux valeurs seuils γ_1 et γ_2.

- $\gamma \geq \gamma_1$: on a un opérateur de type conjonctif
- $\gamma \leq \gamma_2$: on a un opérateur de type disjonctif
- $\gamma_2 \leq \gamma \leq \gamma_1$: on a un opérateur de compromis

Le choix du paramètre γ ne peut s'obtenir que par tâtonnement ou par apprentissage, il n'a pas de signification intuitive qui permettrait de l'obtenir par expertise.

B) Opérateurs à attitude variable

Nous allons maintenant aborder les opérateurs d'agrégation ***à attitude variable***, c'est-à-dire des opérateurs qui se comportent différemment selon les valeurs des évaluations marginales. Suivant les domaines des évaluations marginales, ils peuvent agir comme des opérateurs conjonctifs, de compromis ou disjonctifs. Il n'existe pas de classement entre ces différents opérateurs, on peut simplement les caractériser par quelques propriétés.

a) Sommes symétriques

Les sommes symétriques s sont des opérateurs d'agrégation qui doivent vérifier les conditions aux limites (s(0,0)=0 et s(1,1)=1), qui sont monotones, continus et possèdent la propriété d'autodualité [Silvert 79] :

$$1-s(x,y)=s(1-x,1-y)$$

Toutes les sommes symétriques peuvent s'exprimer sous la forme [Silvert 79] :

$$s(x,y) = \frac{g(x,y)}{g(x,y) + g(1-x,1-y)}$$ avec g continue, non décroissante, positive et g(0,0)=0.

Les sommes symétriques courantes sont construites en prenant g(x,y)=xy, g(x,y)=x+y-xy, g(x,y)=min(x,y) ou g(x,y)=max(x,y).

Les sommes symétriques associatives (exceptées les médianes) adoptent une attitude différente en fonction des valeurs de x et de y [Bloch 96.1] :

- si max(x,y)<0,5 s est un opérateur conjonctif
- si min(x,y)>0,5 s est un opérateur disjonctif
- sinon s est un opérateur de compromis

Un tel opérateur de somme symétrique adopte une attitude pessimiste lorsque les évaluations marginales sont mauvaises, une attitude optimiste si elles sont bonnes et sinon une attitude intermédiaire. Si g est associative, on peut l'étendre à n critères et on construit avec la fonction g étendue l'extension de la somme symétrique à n critères. De même on peut pondérer une somme symétrique s'il existe une version pondérée de la fonction g. On peut par exemple construire une somme symétrique pondérée à partir du minimum pondéré.

Les sommes symétriques associatives et strictement croissantes s'écrivent sous une forme encore simplifiée [Dubois 88.2] :

$s(a_1,\ldots,a_n) = f^{-1}\left(\sum_{i=1}^{n} w_i f(a_i)\right)$ avec f strictement monotone, f(0) et f(1) non bornées et f(1-x)+f(x)=0.

b) Opérateurs MICA

Une dernière catégorie d'opérateurs à attitude variable sont les opérateurs de type MICA (**M**onotonic **I**dentity **C**ommutative **A**ggregator) [Yager 94]. Ce sont des opérateurs continus, monotones mais surtout leur particularité est d'avoir un élément identité g tel que :

$M^n_g(a_1,\ldots,a_n)=M_g^{n+1}(a_1,\ldots,a_n,g)$

On distingue deux catégories d'opérateurs MICA : les FIMICA (**F**ixed **I**dentity **MICA**), pour lesquels l'élément identité g est fixé, et les SIMICA (**S**elf **I**dentity **MICA**) pour lesquels g est le résultat de l'agrégation $M_g(a_1,\ldots,a_n)$.

g joue le rôle d'un élément "neutre" de l'agrégation. Un élément plus petit que g a un poids négatif dans l'agrégation alors qu'un élément plus grand que g a lui un poids positif. Les t-normes sont des FIMICA avec g=1 et les t-conormes des FIMICA avec g=0.

Kelman [Kelman 96] a proposé une classe de FIMICA simples. Ils sont de la forme :

$M_g(a_1,\ldots,a_n) = f\left(\sum_{i=1}^{n} h(a_i)\right)$ avec f et h non décroissantes et h(g)=0.

$h(a_i)$ représente le poids de l'élément a_i dans l'agrégation et la fonction f permet de ramener le résultat de l'agrégation sur le domaine [0,1]. Si on choisit une fonction linéaire pour h, h(x)=x-g, on obtient un opérateur FIMICA :

$$M_g(a_1,\ldots,a_n) = \min\left[1, \max\left(0, g + \sum_{i=1}^{n}(a_i - g)\right)\right]$$

Le choix du paramètre g permet d'adapter le comportement de l'opérateur. g représente le seuil entre les "bons" et les "mauvais" éléments. On remarque d'ailleurs que :

$M_g(a_1,\ldots,a_n)=M_{1-g}(1-a_1,\ldots,1-a_n)$.

IV.3.2 Interprétation des opérateurs

L'objectif de ce chapitre est de caractériser de façon simple l'attitude de chacun des opérateurs en interprétant leurs différentes propriétés [Eude 98.1]. Ceci nous permettra de sélectionner un unique opérateur d'agrégation pour chaque type d'attitude. On reprend la même classification des opérateurs que précédemment.

A) Opérateurs à attitude constante

Les opérateurs à attitude constante, contrairement aux opérateurs à attitude variable, adoptent le même comportement quelles que soient les valeurs à agréger. Dans cette catégorie, chaque opérateur pourra donc être qualifié par une attitude unique.

a) Opérateurs conjonctifs

Les opérateurs conjonctifs sont des opérateurs qui adoptent toujours une attitude pessimiste. Une agrégation conjonctive est au mieux égale à la plus faible des évaluations marginales. Une seule évaluation faible suffit pour donner une valeur d'agrégation faible.

On distingue en fait le cas de l'opérateur ***minimum*** où l'agrégation prend effectivement la valeur de la plus faible des évaluations marginales, et le cas où l'agrégation est strictement inférieure. On choisit l'opérateur ***produit*** comme unique représentant de cette dernière catégorie. Il est en effet facile à calculer et moins excessif que la t-norme de Weber.

Les opérateurs conjonctifs sont les plus sévères. Le choix d'un tel opérateur dans un gabarit signifie que l'on doit reconnaître absolument toutes les sous-activités. Dans le cas du ***minimum***, une seule sous-activité compte, la plus faiblement reconnue, et les autres n'ont pas d'influence. Avec le ***produit*** en revanche, toutes peuvent contribuer au résultat de l'agrégation. On parle d'effet de renforcement.

Reprenons le schéma de décomposition de l'évaluation d'un gabarit (cf. Fig. 71). Avec un ***produit*** on pourra distinguer par exemple les cas où B et C sont reconnues avec les degrés (0,2;0,2) et (0,2;1), alors qu'on ne les distingue pas avec un ***minimum***. Il se peut que C soit présente dans plusieurs gabarits, et sa reconnaissance à elle seule n'est alors pas significative pour le gabarit A. Dans ces conditions, il vaut mieux ne pas distinguer les deux cas précédents pour éviter de partir sur une fausse piste. C'est l'association des activités B et C qui est caractéristique mais pas chacune prise séparément. On peut aussi rencontrer le cas où C à elle seule constitue un indice sérieux tout en nécessitant une confirmation par la reconnaissance de B ; on choisit alors le ***produit***.

b) Opérateurs disjonctifs

Les opérateurs disjonctifs sont duaux des opérateurs conjonctifs. Ils adoptent une attitude optimiste au pire égale à la plus forte des évaluations marginales. Pour les mêmes raisons que précédemment, on gardera deux représentants de cette catégorie, l'opérateur ***maximum*** qui ne prend en compte que la plus grande des évaluations marginales et la ***somme probabiliste***, duale du produit et qui tient compte de tous les éléments.

Le choix d'une disjonction indique que n'importe laquelle des sous-activités est à elle seule significative de l'activité globale. Le choix entre les deux opérateurs s'effectue suivant que la reconnaissance d'une autre sous-activité peut confirmer ou non la reconnaissance globale.

c) Opérateurs de compromis

Les opérateurs de compromis adoptent des attitudes intermédiaires entre l'optimisme et le pessimisme. On observe pour tous ces opérateurs un effet compensatoire plus ou moins prononcé : une évaluation marginale faible peut être compensée par une évaluation marginale forte. Ceci permet de tenir compte du fait que certaines sous-activités sont difficiles à reconnaître mais qu'elles ne doivent pas trop pénaliser la reconnaissance du gabarit si l'on a par ailleurs d'autres indices.

i) Moyennes classiques

Pour la moyenne arithmétique, la compensation est totale. Si toute évaluation marginale faible a est compensée par une évaluation marginale forte 1-a alors le résultat de l'agrégation est moyen et vaut 0,5.

Pour la moyenne géométrique, la compensation n'est pas totale et le résultat pour les mêmes évaluations marginales est strictement inférieur à 0,5. Inversement pour la moyenne quadratique. Nous gardons donc ces 3 opérateurs, chacun étant caractérisé par un niveau de compensation (faible, totale, forte).

ii) Médianes

Pour les médianes, le niveau de compensation est variable. L'intérêt des médianes est généralement de fournir une valeur d'agrégation proche des évaluations marginales. Ceci est vrai si l'on agrège un nombre impair d'éléments. Dans notre cas, cette proximité n'a aucune importance et les médianes n'apportent rien de plus que les moyennes sur le plan sémantique, donc nous excluons ces opérateurs.

iii) OWA

Les OWA sont eux aussi des opérateurs de compensation, mais ils ont la particularité de permettre des agrégations partielles. L'agrégation peut ne tenir compte que d'une certaine proportion de critères à satisfaire. Ce sont les seuls opérateurs qui permettent de traduire des expressions comme "satisfaire au moins la moitié des critères". On peut les utiliser dès que l'expert qualifie l'agrégation avec ce type de quantificateurs linguistiques. Il suffit alors de traduire le quantificateur par un ensemble flou pour trouver l'opérateur OWA correspondant.

d) Intégrales floues

Les intégrales floues permettent de prendre en compte à la fois des poids individuels sur les critères qui reflètent l'importance relative de chaque critère mais aussi des poids sur des groupes de critères traduisant des interactions, des incompatibilités… Mais la construction de ces poids ne se fait pas de façon intuitive et il serait difficile de les extraire de l'expertise.

De plus, les OWA offrent déjà la possibilité d'effectuer des agrégations partielles pondérées. Sans les intégrales floues, la seule chose que l'on ne peut pas exprimer est l'interaction ou l'incompatibilité entre critères et ceci n'apparaît pas dans les besoins pour les gabarits. Nous excluons donc les intégrales floues.

e) Opérateurs de compensation

Les opérateurs de compensation peuvent eux aussi changer d'attitude. Ce changement dépend des évaluations marginales mais aussi du paramètre γ. Mais il est beaucoup plus difficile que précédemment de déterminer exactement les différents comportements adoptés. On a vu en effet que leur attitude dépend de la position de γ par rapport à deux valeurs seuils γ_1 et γ_2 qui elles dépendent des évaluations marginales. Ces deux valeurs seuils n'ont pas une expression simple, elles sont en logarithme. D'autre part, le choix du paramètre γ ne peut s'obtenir que par tâtonnement ou par apprentissage, il n'a pas de signification intuitive qui permettrait de l'obtenir par expertise. On exclut donc ce type d'opérateur.

B) Opérateurs à attitude variable

Les comportements des opérateurs à attitudes variables se définissent par rapport aux attitudes des opérateurs précédents. Selon leurs paramètres, ils peuvent en effet agir tantôt comme l'un, tantôt comme l'autre. Nous allons analyser pour chacun d'eux les différentes attitudes adoptées et essayer de leur donner une interprétation.

a) Sommes symétriques

Les sommes symétriques vérifient toutes une même propriété, l'autodualité. Cette propriété impose que l'opérateur d'agrégation change de comportement lorsque les évaluations marginales prennent la valeur 0,5 : si le comportement est conjonctif pour $x<0,5$ et $y<0,5$ alors il sera disjonctif pour $x>0,5$ et $y>0,5$ et réciproquement.

1. si $\forall(x,y)$ / $x\leq 0,5$ et $y\leq 0,5$ $s(x,y)\leq \min(x,y)$ alors $\forall(X,Y)$ / $X\geq 0,5$ et $Y\geq 0,5$, $1-X\leq 0,5$ et $1-Y\leq 0,5 \Rightarrow s(1-X,1-Y)\leq \min(1-X,1-Y) \Rightarrow s(X,Y)\geq \max(X,Y)$

2. si $\forall(x,y)$ / $x\leq 0,5$ et $y\leq 0,5$ $s(x,y)\geq \max(x,y)$ alors $\forall(X,Y)$ / $X\geq 0,5$ et $Y\geq 0,5$, $1-X\leq 0,5$ et $1-Y\leq 0,5 \Rightarrow s(1-X,1-Y)\geq \max(1-X,1-Y) \Rightarrow s(X,Y)\leq \min(X,Y)$

On trouve des exemples de sommes symétriques pour les deux cas de comportements. On est dans le cas 1 en construisant la somme symétrique à partir de $g(x,y)=xy$ et dans le cas 2 pour $g(x,y)=x+y-xy$. On peut aussi trouver des sommes symétriques dont le comportement varie sur une plage plus restreinte et est toujours celui d'un opérateur de compromis. Avec $g(x,y)=\min(x,y)$, on passe d'un comportement de compromis faible à un compromis fort et inversement avec $g(x,y)=\max(x,y)$.

Certaines sommes symétriques adoptent une attitude plutôt pessimiste lorsque les sous-activités sont faiblement reconnues et plutôt optimiste lorsqu'elles le sont fortement et réciproquement. Dans un cas on atténue la force ou la faiblesse des reconnaissances partielles (attitude prudente) et dans l'autre on les accentue (attitude engagée).

Les notions de valeurs faibles et fortes se définissent toujours par rapport à la valeur moyenne 0,5. C'est pourquoi nous avons voulu créer une famille d'opérateurs d'agrégation ayant le même type de comportement que les sommes symétriques mais paramétrable par un réel g qui serait la frontière de changement de comportement et représenterait le seuil entre les valeurs faibles et fortes. Nous verrons comment étendre les sommes symétriques pour obtenir un tel résultat.

b) Opérateurs MICA

Les opérateurs MICA sont paramétrés par un élément g réel de [0,1] dit "neutre". Toute sous-activité reconnue avec un degré supérieur à g renforce positivement l'agrégation et inversement. Si l'on agrège une infinité de "mauvais" éléments on tend vers une valeur d'agrégation nulle et si l'on agrège une infinité de bons éléments on tend vers la valeur d'agrégation maximale 1. Il y a renforcement.

Les opérateurs uninormes proposés par Yager [Yager 96] sont intéressants car leur attitude peut être caractérisée facilement. La classe des uninormes est équivalente à la classe des FIMICA associatifs. Les uninormes passent d'une attitude conjonctive lorsque toutes les évaluations sont inférieures à l'élément neutre à une attitude disjonctive lorsque toutes les évaluations sont supérieures à l'élément neutre. En d'autres termes, si tous les éléments sont mauvais le résultat de l'agrégation est inférieur au minimum des évaluations et il est supérieur au maximum des évaluations si tous les éléments sont bons.

Cette famille d'opérateurs a un comportement tout à fait comparable aux sommes symétriques du cas 1, simplement la frontière entre bons et mauvais éléments qui marque les changements d'attitude est paramétrée (c'est l'élément neutre de l'opérateur) au lieu d'être fixée à 0.5.

Un cas particulier d'uninorme pondérée est l'opérateur introduit par Kelman [Kelman 96], définit par :

$$M_g(a_1,\ldots,a_n) = g + n\sum_i w_i(a_i - g)$$

IV.3.3 Extension des sommes symétriques

A) Motivations

Les opérateurs uninormes apparaissent donc comme une extension des sommes symétriques de cas 1. Ils correspondent à des sommes symétriques à comportement pessimiste avec les valeurs faibles et optimiste avec les valeurs fortes. Mais ils apportent une souplesse en paramétrant la limite entre bons et mauvais éléments.

Il n'existe pas de telle extension pour les sommes symétriques de cas 2. C'est pour cette raison que nous avons étendu toutes les sommes symétriques en des opérateurs de comportements similaires mais paramétrables par un réel g [Eude 98.1]. D'où le nom de ***"sommes g-symétriques"***. Cette nouvelle famille englobe à la fois les uninormes, extensions des sommes symétriques de cas 1, plus une extension des sommes symétriques de cas 2.

B) Définition

La définition minimal d'un opérateur d'agrégation est rappelée en IV.3.1 p.150. Cette définition impose quelques propriétés. Les sommes g-symétriques se définissent en ajoutant à ces propriétés une extension de la propriété d'autodualité. Au lieu d'avoir $s(1-a_1,\ldots,1-a_n)=1-s(a_1,\ldots,a_n)$, ce qui donne une symétrie par rapport à $(0{,}5,\ldots,0{,}5)$, on pose : $s(2g-a_1,\ldots,2g-a_n)=2g-s(a_1,\ldots,a_n)$, ce qui donne une symétrie par rapport à $(g,\ldots,g)$.

Définition : une somme g-symétrique est une fonction vérifiant :

1. $s_g : [0,1]^n \rightarrow [0,1]$

2. s_g continue

3. s_g non décroissante en chacun de ses arguments

4. $g \in [0,1]$

5. $\forall\ (a_1,\ldots,a_n) \in [\max(0,2g-1),\min(1,2g)]^n,\ s_g(2g-a_1,\ldots,2g-a_n)=2g-s_g(a_1,\ldots,a_n)$

C) Propriétés

Soit $g\in[0,1]$ et une fonction f définie sur $[2g-1,2g]^n\cup[0,1]^n$, continue, vérifiant :

1. $\forall(a_i), f(a_i) \neq -f(2g-a_i)$

2. Pour chaque paramètre X : $\dfrac{\partial f}{\partial X}(a_i)\times f(2g-a_i)+\dfrac{\partial f}{\partial X}(2g-a_i)\times f(a_i)\geq 0$.

Propriété 1 : la fonction s_g définie comme suit est une somme symétrique.

$$s_g(a_i)=\max\left\{0,\min\left[1{,}2g\frac{f(a_i)}{f(a_i)+f(2g-a_i)}\right]\right\} \qquad \textbf{(1)}$$

Réciproque (Propriété 1') : Toute somme symétrique s_g peut s'écrire sous la forme **(1)**.

Démonstration : on peut prendre par exemple $f=s_g$ sur $[0,1]^n$ et $f(a_i)=2g-s_g(2g-a_i)$ sur $[2g-1,0]^n$ (resp. $[1,2g]$) si $g<0{,}5$ (resp. $g>0{,}5$). Les propriétés précédentes de f sont bien vérifiées, l'expression aux dérivées devenant :

$$\frac{\partial s_g}{\partial X}(a_i)\times\left(2g-s_g(a_i)\right)+\frac{\partial s_g}{\partial X}(a_i)\times s_g(a_i)=2g\frac{\partial s_g}{\partial X}(a_i)\text{ avec }\frac{\partial s_g}{\partial X}\geq 0.$$ ❐

Suivant les propriétés de f, on peut obtenir certaines propriétés supplémentaires sur s_g.

Propriétés :

2. si f est commutative, s_g est commutative.

3. si g est élément neutre de f, g est aussi élément neutre de s_g.

4. si g est élément absorbant de f, g est aussi élément absorbant de s_g.

D) Interprétation

Nous allons voir que le paramètre g joue effectivement un rôle de frontière dans le changement d'attitude d'une somme g-symétrique s. On étudie plus précisément les comportements suivants :

1. si $[\forall\ i\ a_i<g \Rightarrow s(a_1,\dots,a_n)>\max(a_i)]$ alors
   $[\forall\ i\ a_i>g,\ s(a_1,\dots,a_n)=s(2g-b_1,\dots,2g-b_n)=2g-s(b_1,\dots,b_n)$ avec $b_i<g$ donc
   $s(b_1,\dots,b_n)>\max(b_i) \Rightarrow s(a_1,\dots,a_n)<2g-\max(b_i) \Rightarrow s(a_1,\dots,a_n)<\min(a_i)]$

2. si $[\forall\ i\ a_i<g \Rightarrow s(a_1,\dots,a_n)<\min(a_i)]$ alors $[\forall\ i\ a_i>g \Rightarrow s(a_1,\dots,a_n)>\max(a_i)]$.

3. si $[\forall\ i\ a_i<g \Rightarrow \min(a_i)<s(a_1,\dots,a_n)<1/n\ \Sigma(a_i)]$ alors
   $[\forall\ i\ a_i>g \Rightarrow 1/n\ \Sigma(a_i)<s(a_1,\dots,a_n)<\max(a_i)]$.

4. si $[\forall\ i\ a_i<g \Rightarrow 1/n\ \Sigma(a_i)<s(a_1,\dots,a_n)<\max(a_i)]$ alors
   $[\forall\ i\ a_i>g\ \min(a_i)<s(a_1,\dots,a_n)<1/n\ \Sigma(a_i)]$.

Les comportements 1 et 4 correspondent à une attitude optimiste pour les valeurs faibles (par rapport à g) et pessimiste pour les valeurs fortes (par rapport à g), le comportement 1 étant tout de même plus extrême que le 4. Les comportements 2 et 3 sont duaux des 1 et 4.

On a donc 2 catégories de sommes g-symétriques, de même que l'on avait 2 catégories de sommes symétriques. La première catégorie d'opérateur tend à ramener les évaluations vers une valeur moyenne g alors que la deuxième tend à l'en écarter. Dans un cas on accentue les différences alors que dans l'autre on les minimise. Les deux catégories d'opérateurs peuvent être utiles selon ce que l'on veut exprimer.

La première catégorie dénote une attitude prudente. On ne veut pas négliger un gabarit s'il y a le moindre indice même faible, mais en revanche on n'est pas certain de reconnaître le gabarit même si tous les indices sont forts. On peut adopter une telle attitude avec un gabarit représentant une menace par exemple : on souhaite rester vigilant sur l'éventualité de cette menace, mais on attend d'avoir bien reconnu tous les éléments avant de prendre une décision en réaction à cette menace.

La deuxième catégorie donne une attitude plus extrémiste. Elle peut être adoptée par exemple dans un cas où la reconnaissance des sous-activités est très bruitée, c'est-à-dire qu'une reconnaissance faible peut être liée à une erreur et on ne veut pas que cela ait d'impact sur la reconnaissance du gabarit. En revanche, dès que la reconnaissance d'une sous-activité dépasse un certain seuil (correspondant à g), on estime qu'on peut en tenir compte, et une seule sous-activité est significative donc l'agrégation est disjonctive.

Dans ces deux cas, g peut s'interpréter un peu différemment. Il correspond toujours à la limite entre des valeurs faibles et fortes. Lorsqu'il n'est pas question de bruit, g dépend simplement de la difficulté que l'on a à reconnaître les activités. Plus les activités sont difficiles à reconnaître, plus on attache d'importance à des valeurs de reconnaissance faibles et donc plus g est faible. En revanche, si la reconnaissance est bruitée, g représente la limite entre les valeurs non significatives du fait du bruit et les valeurs significatives.

On considère maintenant une somme g-symétrique s(x,y) prenant seulement 2 arguments. Pour plus de 2 arguments, on peut prendre $s(a_1,\dots,a_n)=s[\min(a_i),\max(a_i)]$. Pour une telle fonction on peut passer d'une catégorie à l'autre par une transformation. Examinons la fonction :

$$h_g(x,y) = \max\left\{0, \min\left[1, \min(x,y) + \max(x,y) - s_g(x,y)\right]\right\} \qquad (2)$$

Si h_g est bien non décroissante, h_g est une somme g-symétrique. Si s_g adopte l'attitude 1 (resp. 3), h_g aura l'attitude 2 (resp. 4) et réciproquement. Cette transformation peut donc servir pour la construction de nouvelles sommes g-symétriques.

E) Exemples

A titre d'exemple, nous proposons des sommes g-symétriques de catégorie 1 inspirées des sommes symétriques classiques. Elles sont construites à partir de l'équation (1) p.163. On a simplement ajouté des constantes aux fonctions génératrices pour que le dénominateur soit non nul. En fait, on a choisit des fonctions génératrices f telles que $f(x,y)\geq 0$ et $f(x,y)+f(2g-x,2g,y)\geq 0$ pour $(x,y)\in[0,1]^2$.

On génère deux sommes g-symétriques g_1 et g_2 d'attitude 1 à partir de :

$f_1(x,y)=x+y-xy+(1-g)^2$ et

$f_2(x,y)=\max(x,y)+| 1-2g|$.

g_1 et g_2 se ressemblent sauf pour les valeurs extrêmes de g, de x et de y. On préférera la fonction g_1 moins extrémiste dans le cas de valeurs extrêmes.

On obtient deux sommes g-symétriques g_3 et g_4 d'attitude 2 en appliquant la transformation (2) sur g_1 et g_2 :

$g_3(x,y)=\min(x,y)+\max(x,y)-g_1(x,y)$ et

$g_4(x,y)=\min(x,y)+\max(x,y)-g_2(x,y)$

Il n'y a quasiment pas de différences entre ces deux fonctions. On utilise prioritairement g_4, plus facile à calculer.

L'opérateur FIMICA de Kelman est peu différent de g_3 et de g_4 pour deux arguments mais la différence se fait pour plus de deux arguments. Avec l'opérateur FIMICA toutes les évaluations comptent pour l'agrégation alors que pour g_3 ou g_4 seules les évaluations extrêmes sont considérées.

Le comportement de ces fonctions ainsi que celui des sommes symétriques, des opérateurs de compensation et d'un opérateur FIMICA est illustré en annexe VII.8 p.260.

IV.3.4 Tableau de synthèse

Voici un tableau résumant les attitudes des différents opérateurs. On donne pour chaque opérateur à attitude variable la plage de ses variations par rapport aux opérateurs à attitude fixe et suivant les valeurs des évaluations x et y. Ce tableau intègre les sommes g-symétriques :

<table>
<tr><th></th><th>ET-</th><th>ET</th><th>COMPROMIS-</th><th>MOYENNE</th><th>COMPROMIS+</th><th>OU</th><th>OU+</th></tr>
<tr><td>attitude

constante</td><td>produit

xy</td><td>minimum

min(x,y)</td><td>moyenne géométrique
$\sqrt{xy}$</td><td>moyenne arithmétique
(x+y)/2</td><td>moyenne quadratique
$\sqrt{(x^2+y^2)/2}$</td><td>maximum

max(x,y)</td><td>somme probabiliste
x+y-xy</td></tr>
<tr><td>agrégation partielle</td><td colspan="2"></td><td colspan="3">OWA ↔
$\sum(p_i a_{(i)})$</td><td colspan="2"></td></tr>
<tr><td rowspan="4">sommes

symétriques

$\frac{g(x,y)}{g(x,y)+g(1-x,1-y)}$</td><td colspan="7">s1 construite avec g(x,y)=xy →
x+y=1</td></tr>
<tr><td colspan="2"></td><td colspan="4">s2 construite avec g(x,y)=min(x,y) →
x+y=1</td><td></td></tr>
<tr><td colspan="7">← s3 construite avec g(x,y)=x+y-xy
x+y=1</td></tr>
<tr><td colspan="2"></td><td colspan="3">← s4 construite avec g(x,y)=max(x,y)
x+y=1</td><td colspan="2"></td></tr>
<tr><td>uninormes</td><td colspan="7">$M_g(a_i)=g+n\Sigma w_i(a_i-g)$ →
x=g,y=g</td></tr>
<tr><td rowspan="4">sommes

g-symétriques

$\frac{2gf(x,y)}{f(x,y)+f(2g-x,2g-y)}$</td><td colspan="7">← g1 construite avec f1(x,y)=x+y-xy+(1-g) 2
x+y=2g</td></tr>
<tr><td colspan="7">← g2 construite avec f2(x,y)=max(x,y)+|1-2g|
x+y=2g</td></tr>
<tr><td colspan="7">g3(x,y)=x+y-g1(x,y) →
x+y=2g</td></tr>
<tr><td colspan="7">g4(x,y)=x+y-g2(x,y) →
x+y=2g</td></tr>
</table>

Fig. 88 : tableau de synthèse des opérateurs d'agrégation

IV.3.5 Choix d'un opérateur d'agrégation

Cette analyse nous permet maintenant d'établir un processus d'aide à la décision pour le choix d'un opérateur d'agrégation. Ce choix se fait en fonction des propriétés de comportement que l'on souhaite obtenir. Cette procédure est décrite sous la forme d'un arbre (Fig. 89) où chaque branche correspond à un choix de propriété. Lorsque l'on aboutit à une feuille terminale de l'arbre, l'opérateur d'agrégation est déterminé.

Les choix qui ne se justifient que pour des agrégations de plus de deux critères sont en pointillés. Si l'on dispose de poids sur les sous-activités, on utilise l'extension avec pondération des opérateurs si elle existe. Les opérateurs pour lesquels il existe une telle extension sont soulignés. Ceci supprime le choix entre renforcement et non renforcement pour les opérateurs conjonctifs et disjonctifs. Le choix permis est donc un peu moins subtil à ce niveau mais ceci est compensé par la prise en compte d'importances relatives.

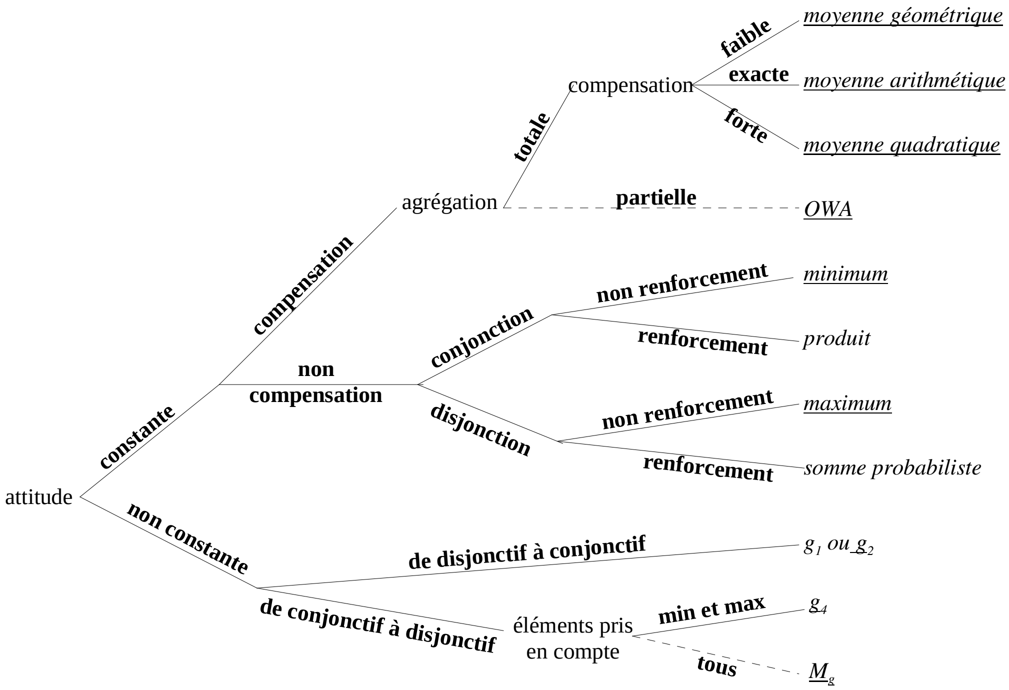


Fig. 89: choix d'un opérateur d'agrégation

IV.4 Conclusion

Pour l'application à la technique des gabarits, nous avons besoin d'opérateurs d'agrégation pour réaliser deux opérations différentes. D'une part on souhaite agréger en chaque nœud du gabarit les compatibilités par rapport à chaque contrainte pour calculer une compatibilité partielle, d'autre part on agrège les compatibilités partielles pour obtenir un degré de compatibilité globale qui correspond à la certitude de l'hypothèse de gabarit.

Considérons d'abord l'agrégation des compatibilités des différentes contraintes. Les contraintes peuvent éventuellement avoir différentes importances relatives, l'agrégation peut donc être pondérée. On peut en effet estimer que le critère portant sur la durée d'une activité est plus important que sa date de début ou inversement.

Un élément particulier d'un gabarit est reconnu si les indices perçus permettent de vérifier toutes ses contraintes. Il suffit qu'une contrainte ne puisse pas être vérifiée pour que l'élément ne soit pas reconnu. Prenons l'exemple où l'on attend un événement de type A de durée d et de date de fin f. Les indices permettent de reconnaître un événement A' de type A de durée tout à fait compatible avec d mais pour lequel la date de fin f est très peu probable. A' peut ne pas correspondre du tout à l'événement attendu A. C'est la contrainte la moins compatible qui compte, il suffit qu'une contrainte ne soit pas vérifiée pour que l'événement soit totalement différent.

On utilisera donc une agrégation conjonctive du type ***minimum,*** éventuellement pondérée. C'est ce qui est utilisé dans la plupart des systèmes de satisfaction de contraintes lorsque la vision égalitariste est adoptée : le niveau de satisfaction du problème complet est celui de la contrainte la moins satisfaite, et si une contrainte est violée le système est non satisfait.

Pour l'agrégation des compatibilités partielles en revanche, le raisonnement peut être totalement différent. Le schéma de décomposition d'un gabarit en sous-activités traduit le mécanisme de reconnaissance de l'opérationnel. Si l'activité A se décompose en B et C, cela peut avoir plusieurs significations. A peut être reconnue si B et C sont reconnues ou seulement si l'une des deux activités l'est, suivant que les activités B et C sont caractéristiques ou non. Si l'activité C intervient dans de nombreux gabarits, sa reconnaissance sera beaucoup moins déterminante pour la reconnaissance de A.

Le schéma de décomposition d'un gabarit est sensé traduire le mécanisme de reconnaissance de l'opérationnel. Chaque agrégation est l'expression d'une heuristique particulière dans le raisonnement de l'officier de renseignements. Le choix de l'agrégation ne peut donc être déterminé que par lui et en fonction de la signification qu'il donne à la décomposition. L'arbre de décision lui permet de décomposer cette procédure de choix en faisant ressortir les critères importants.

De même c'est l'opérationnel qui fixe les poids relatifs éventuels des éléments à agréger. Pour cela on peut utiliser des termes linguistiques ("important", "peu important"…) plus faciles à manipuler que des valeurs numériques. Il fixe aussi le paramètre g dans le cas de sommes g-symétriques. Dans ce cas l'interprétation de g dépend de l'utilisation de l'opérateur.

V Reconnaissance de scenarii spatiaux

Nous avons vu jusqu'ici comment traiter les aspects temporels pour la reconnaissance d'activités. Mais dans certains gabarits, comme le gabarit ***entraînement sur site*** présenté en figure Fig. 4 p.9, les aspects spatiaux sont aussi importants. On peut imaginer aussi des gabarits où les contraintes spatiales sont prépondérantes sur les contraintes temporelles. Ce peut être le cas si l'on cherche à reconnaître une configuration particulière des unités sur le terrain sans s'occuper de leur progression. Il a aussi été envisagé d'utiliser les gabarits pour de la reconnaissance de photos aériennes. Dans tous ces cas les contraintes spatiales deviennent essentielles.

On retrouve beaucoup de préoccupations communes dans les modélisations temporelles et spatiales. Dans les deux cas on a des contraintes qui peuvent être exprimées de façon qualitative ou quantitative. Les informations peuvent être partielles, imparfaites, et être exprimées à des niveaux de granularité différents. Et dans les deux cas on cherche à effectuer des inférences. L'analogie entre le couple (position, distance) et le couple (date, durée) est flagrante. Connaissant la date de A et la durée entre A et B on déduit la date de B. De même connaissant la position de A et la distance entre A et B on cherche la position de B.

La différence essentielle entre temporel et spatial tient dans la dimension de ces deux aspects. Le temps est à une dimension alors que l'espace est à deux ou trois dimensions, ce qui complique nettement le traitement des aspects spatiaux. Les différents aspects spatiaux que l'on peut considérer sont aussi plus nombreux : topologie, forme, distance, direction…

Beaucoup de travaux sur le spatial s'inspirent donc de ce qui est fait sur le temporel, et les raisonnements font appel aux même techniques. Nous allons voir quelles sont les différentes modélisations spatiales proposées dans la littérature. Par rapport aux besoins de représentation dans les gabarits, nous verrons ce que peuvent apporter ces différentes modélisations et aussi ce qui leur manque. De là nous construirons un modèle fondé sur les FCSP, par extension de ce qui est fait pour la reconnaissance temporelle. Nous verrons que ce modèle augmente le pouvoir d'expression par rapport à ceux existants. Enfin nous détaillerons les algorithmes de raisonnement, plus complexes. Ces algorithmes ne sont pas complets comme dans le cas temporel, mais ils permettent déjà un raisonnement beaucoup plus poussé que la simple vérification des contraintes qui était faite jusqu'ici.

V.1 Modélisations spatiales

Nous présentons ici les différentes modélisations spatiales existantes et les types de raisonnement qu'elles permettent. Il existe en fait deux grandes catégories de modèles, les modèles spatiaux numériques et les modèles spatiaux qualitatifs. Nous nous intéressons en fait à ceux qui permettent de modéliser l'imprécision, numériquement ou qualitativement.

Les modèles numériques classiques sont des modèles vecteur ou matriciel (aussi appelé "raster"). Dans le modèle matriciel, l'espace est découpé régulièrement en grilles de points. Une grille établit la correspondance entre un point de l'espace, dont on connaît les coordonnées dans un référentiel, et une donnée numérique (altitude du point par exemple) ou symbolique (objet de type route par exemple). Le modèle vecteur modélise les objets à partir de primitives qui sont des points, des arcs et des faces. Un point est défini par sa position, un arc par une liste ordonnée de points et une face par un arc fermé délimitant sa frontière. On peut aussi ajouter des relations entre objets. Ces relations sont géométriques (union, différence, inclusion, chevauchement) et topologiques (connectivité, adjacence, voisinage). Les deux modèles sont numériques au sens où les points sont décrits par leurs coordonnées. Le modèle vecteur autorise cependant certaines relations qualitatives supplémentaires.

Les modèles purement qualitatifs sont plus riches au sens où ils donnent une interprétation beaucoup plus intuitive de la réalité, dans des termes compréhensibles. Ils sont aussi beaucoup plus économiques du point de vue stockage et traitement des informations. Ils n'apportent certes pas la précision des modèles numériques, mais dans de nombreux cas une vue qualitative est largement suffisante et plus parlante.

Les modèles qualitatifs sont surtout utilisés dès que l'on a une description de l'espace imparfaite. L'imperfection se traduit la plupart du temps par des imprécisions mais peut aussi être liée à de l'incomplétude. Les modèles qualitatifs en employant des relations vagues qui recouvrent plusieurs situations rendent bien compte de cette imprécision. De plus, on peut choisir un niveau de précision en utilisant des relations plus ou moins fines.

Nous présentons aussi un certain nombre de modèles ayant les mêmes préoccupations que les modèles qualitatifs : les représentations symboliques, les modèles flous ainsi que tous les modèles numériques intégrant la notion d'imprécision (modèle haze-point cf. V.1.3B) p.206 et modèle potentiel cf. V.1.3A) p.204). Ces modèles ont les avantages des modèles qualitatifs classiques puisqu'ils intègrent l'imprécision des informations et permettent des interprétations qualitatives. Un autre avantage est de permettre l'intégration d'informations purement numériques, ce qui en contrepartie augmente la complexité.

Les descriptions spatiales comportent plusieurs aspects. On peut s'intéresser aux relations topologiques, aux distances, à des positions relatives ou à la forme des objets. On passe en revue chacun de ces aspects pour les différents modèles. Les modèles flous sont souvent obtenus par extension des notions qualitatives.

V.1.1 Modélisations qualitatives

A) Topologie

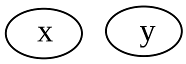
La topologie est un des multiples aspects de l'arrangement des objets dans l'espace. Elle concerne les objets de dimension supérieure ou égale à deux et s'intéresse à la position relative des frontières de ces objets. Une caractérisation des relations topologiques est qu'elles sont invariantes par les transformations topologiques, c'est-à-dire qu'elles sont préservées si l'on applique une translation, une rotation ou un facteur d'échelle sur les objets [Egenhofer 91].

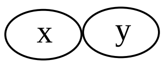
a) RCC

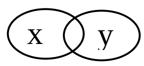
La théorie RCC (Region Connection Calculus) a été développée ces dernières années par une équipe de chercheurs de Leeds, [Cohn&al 93], [Cohn&al 97], [Cui&al 92], [Cui&al 93]). Elle est née du besoin d'avoir une représentation spatiale qualitative, par opposition aux modélisations numériques habituelles. RCC est une tentative pour s'affranchir des aspects quantitatifs et pour rester plus proche du raisonnement humain, de même que la théorie d'Allen dans le domaine temporel.

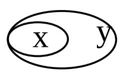
i) Modélisation

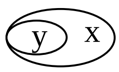
L'élément spatial primitif est une région de l'espace à 2 ou 3 dimensions. Toutes les régions doivent être de la même dimension. Les relations primitives sont définies à partir d'une notion de base, la connexion. Deux régions x et y sont connectées (C(x,y)) si les fermetures topologiques des deux régions possèdent au moins un point d'intersection. Des axiomes précisent les propriétés de C. A partir de C on définit 8 relations spatiales de sens commun [Cui&al 92] :

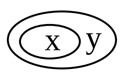
- DC(x,y) : x déconnecté de y 

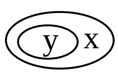
- EC(x,y) : x est connecté extérieurement à y 

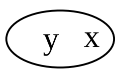
- PO(x,y) : x recouvre partiellement y 

- TPP(x,y) : x est une partie propre tangentielle de y 

- TPPi(x,y) : inverse de TPP(x,y) 

- NTPP(x,y) : x est une partie propre non tangentielle de y 

- NTPPi(x,y) : inverse de NTPP(x,y) 

- EQ(x,y) : x est identique à y 

Ces relations sont exhaustives et disjointes deux à deux. Elles forment la théorie RCC8. On peut raffiner ces relations pour exprimer des notions topologiques plus fines comme la connexité par exemple. Les systèmes RCC15 et RCC23 comportent respectivement 15 et 23 relations primitives. Le niveau de raffinement est fixé par les besoins de l'application et par la complexité de calcul autorisé. C'est l'application qui détermine les situations qu'il est utile ou non de distinguer et donc le nombre de primitives qu'il faut considérer.

ii) Raisonnement

La théorie RCC s'exprime sous la forme d'un calcul de prédicats du premier ordre. La première forme de raisonnement possible est donc le calcul de preuves. Mais il s'avère que le calcul automatique de preuves est peu efficace dans la plupart des cas. De plus on peut démontrer que la théorie RCC est indécidable.

Une méthode de raisonnement plus efficace utilise les tables de transition ou de composition qui avaient déjà été introduites dans la théorie d'Allen. Ces tables donnent pour tout couple de relations primitives $R_1(x,y)$ et $R_2(y,z)$ les relations primitives possibles entre x et z. Ces tables permettent de répondre à des questions du type : "quelle est la relation qui lie les régions A et B ?" ou à détecter des inconsistances dans une base de données en appliquant une vérification de la cohérence pour chaque triplet de régions. On ne peut en général pas assurer la consistance globale par ce seul mécanisme de vérification de cohérence, l'algorithme n'étant pas complet.

Un troisième type de raisonnement introduit un aspect dynamique. L'hypothèse fondamentale est celle de la continuité dans le changement. Le mouvement ainsi que les déformations des régions sont continues, on ne peut pas passer brusquement d'un état à n'importe quel autre. A partir de cette hypothèse on construit des réseaux de continuité qui décrivent les transitions possibles. Voici le réseau de continuité pour RCC8 :

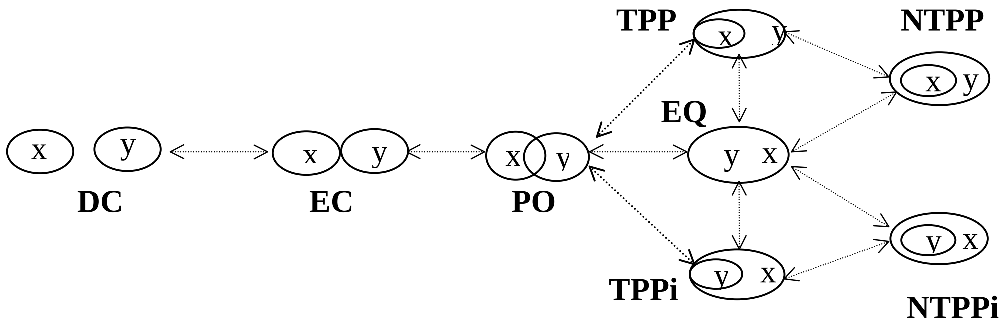


Fig. 90 : graphe des transitions pour RCC8

Ce graphe de transitions permet par exemple d'effectuer le raisonnement suivant : si on a DC(x,y) à un instant t_1 puis PO(x,y) à un instant t_2 ultérieur, alors on peut affirmer qu'il existe un instant t tel que $t_1<t<t_2$ et à cet instant t on a EC(x,y). Les graphes de transition sont utilisés essentiellement pour de la simulation qualitative : à partir d'un état donné, ils fournissent tous les états suivants possibles.

iii) Extension à des régions vagues

Le raisonnement spatial qualitatif peut être vague dans deux cas : les descriptions que l’on a sont imprécises ou les régions que l’on manipule sont par nature imprécises, comme "le sud de l'Angleterre" par exemple. On étend la théorie RCC au cas de régions aux contours vagues [Cohn&al 96.1], [Cohn&al 97].

Dans cette optique on introduit une primitive < qui permet d'ordonner les régions. x<y signifie que x est une région plus précise que y. < est asymétrique, transitive et irréflexive. A partir de cette relation on définit ce qu’est une région précise et on décompose une région vague x en deux régions précises, la plus petite egg(x) et la plus grande yolk(x) extensions possibles de x. C’est la théorie de "l’œuf sur le plat" (egg-yolk).

On part de RCC5, version simplifiée de RCC8, puis on construit les relations primitives entre deux régions vagues x et y en combinant les relations possibles entre egg(x), egg(y), yolk(x) et yolk(y). On aboutit à 46 relations primitives. Voici un exemple de situation :

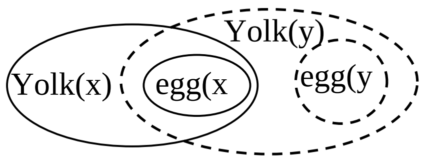


Fig. 91 : exemple de relation entre deux régions vagues

Ces 46 relations primitives peuvent se découper en 13 principaux groupes de relations. Classiquement on utilise ces 13 groupes comme relations primitives entre deux régions vagues.

b) Système 9-intersections

Parallèlement à la théorie RCC s'est développée une autre modélisation spatiale topologique appelée "9-intersections". Ces deux théories sont très proches dans leurs résultats bien que l'approche initiale soit totalement différente. Toute la construction de RCC se fait à partir du prédicat de connexion, alors que, comme on va le voir, la théorie "9-intersections" est fondée essentiellement sur les notions topologiques issues des mathématiques.

i) Construction

Contrairement à la théorie RCC où l'élément de base est la région, le système 9-intersections [Egenhofer 91] se construit en utilisant comme élément de base le point. Il reprend toutes les notions définies en topologie classique sur les ensembles de points. Soit A un ensemble, $\mathring{A}$ est son intérieur, $\overline{A}$ sa fermeture, A^{-1} son complément et $\delta A=\overline{A}\cap\overline{A^{-1}}$ sa frontière.

Les relations entre deux ensembles A et B sont alors déterminées par les intersections entre leurs intérieurs, leurs frontières et leurs compléments. On considère la matrice suivante :

$$I(A,B)=\begin{pmatrix} \partial A\cap\partial B & \partial A\cap\overset{o}{B} & \partial A\cap B^{-1} \\ \overset{o}{A}\cap\partial B & \overset{o}{A}\cap\overset{o}{B} & \overset{o}{A}\cap B^{-1} \\ A^{-1}\cap\partial B & A^{-1}\cap\overset{o}{B} & A^{-1}\cap B^{-1} \end{pmatrix}$$

Il y a 9 intersections à calculer, ces intersections pouvant être vides ou non vides. Il existe 9^2 combinaisons, mais toutes ne sont pas possibles. On trouve 8 combinaisons possibles dans le cas où A et B sont de dimension 2 et ne sont pas des objets complexes (on exclut les objets avec trous par exemple). On construit ainsi 8 relations topologiques, qui se trouvent être identiques aux primitives de RCC8. Le système 4-intersections, plus simple, est obtenu de façon similaire en ne considérant que les intersections de frontières et d'intérieurs.

ii) Raisonnement

Le fondement du raisonnement spatial dans la théorie 9-intersections repose une fois de plus sur l'opération de composition. Le calcul des tables de composition se fait toujours en raisonnant sur les intersections. A partir des matrices d'intersections de R_1 et R_2 et de règles d'inférence du type $A\subseteq B\wedge B\subseteq C\Rightarrow A\subseteq C$ on déduit la matrice d'intersections de $R_1\circ R_2$ et on l'assimile à une disjonction de primitives. On retrouve la même table que dans RCC8.

Un des aspects importants du raisonnement est aussi de prévoir les changements de relations topologiques lorsque l'on applique des transformations sur les objets [Egenhofer&al 92]. On considère ici les transformations topologiques classiques de rotation, translation ou changement d'échelle. Lorsque la même transformation topologique est appliquée simultanément sur tous les objets, par définition les relations topologiques sont inchangées. Mais lorsqu'elles sont appliquées sur un objet unique, on peut observer progressivement des variations dans les relations topologiques. Le problème réciproque consiste à déterminer quel est le type de transformation appliquée sur un objet en fonction des variations de relations topologiques observées.

On calcule une distance topologique entre deux relations à partir de leur représentation matricielle 9-intersections. On construit un graphe où chaque relation primitive est liée à la ou les relations les plus proches suivant cette distance topologique. On retrouve ainsi le graphe de transition de la théorie RCC. Suivant la situation où l'on se trouve il permet de prédire la ou les situations suivantes possibles, et suivant le chemin du graphe emprunté il permet aussi de déduire la ou les types de transformation appliquée. Il existe en effet des chemins caractéristiques pour chaque type de transformation.

iii) Extension à des régions vagues

On peut étendre le système 9-intersections à des régions vagues [Clementini&al 94.2]. Pour cela il faut redéfinir ce qu'est une région. Une région vague A est composée de deux régions simples A_1 et A_2 avec A_1 incluse dans A_2 et δA_1 est la frontière intérieure de A et δA_2 sa frontière extérieure. La frontière vague de la région vague A est définie par $\Delta A = \overline{A_2 - A_1}$. L'intérieur de A est redéfini par $\mathring{A} = A_2 - \Delta A$, le complémentaire par $A^{-1} = E - A_2$, E étant l'espace, et la fermeture par $\overline{A} = \mathring{A} \cup \Delta A$. Les régions simples peuvent être vues comme des cas particuliers de régions vagues où A_1=A_2.

Une fois ces nouvelles définitions posées, on peut recalculer la matrice 9 intersections en remplaçant la frontière par la frontière vague. On trouve ainsi 40 relations regroupées en 14 catégories, ce qui est légèrement différent du résultat obtenu dans l'extension de RCC à des régions vagues.

iv) Méthode "dimension-extended"

Jusqu'ici nous n'avons considéré que des relations topologiques entre régions de même dimension. Mais on peut étendre ces systèmes à 6 groupes de relations : relations surface/surface, surface/ligne, surface/point, ligne/ligne, ligne/point et point/point [Clementini&al 94.1], [Clementini&al 95]. On recalcule la matrice d'intersection en prenant en compte la dimension de l'intersection, d'où le nom de méthode "dimension-extended". La dimension de l'intersection est soit vide, soit nulle (un point), soit 1 (une ligne), soit 2 (une surface). On obtient 52 relations distinctes en se plaçant dans le système 4-intersections.

c) Méthode "calculus-based"

On souhaite ici construire un système de relations topologiques applicables comme précédemment sur des objets de dimensions différentes de type point, ligne ou surface. L'inconvénient des méthodes 4-intersections, 9-intersections ou "dimension-extended" est qu'elles conduisent à un grand nombre de relations dont beaucoup sont peu intuitives. Elles sont par conséquent difficilement manipulables.

La méthode "calculus-based" est, elle, construite à partir de la définition de 5 relations de base très intuitives et d'opérateurs de calcul de frontière [Clementini&al 94.1], [Clementini&al 95].

Voici les définitions (λ_i est un objet de type surface, ligne ou point) :

1. Relation **touche** (ne s'applique pas entre points) :

   $(\lambda_1 \text{touche } \lambda_2) \Leftrightarrow (\overset{o}{\lambda}_1 \cap \overset{o}{\lambda}_2 = \emptyset) \wedge (\lambda_1 \cap \lambda_2 \neq \emptyset)$

2. Relation **dans** : $(\lambda_1 \text{ dans } \lambda_2) \Leftrightarrow (\overset{\circ}{\lambda}_1 \cap \overset{\circ}{\lambda}_2 \neq \emptyset) \wedge (\lambda_1 \cap \lambda_2 = \lambda_1)$

3. Relation **traverse** (relation ligne/ligne et ligne/surface) :

   $(\lambda_1 \text{traverse } \lambda_2) \Leftrightarrow \dim(\overset{o}{\lambda}_1 \cap \overset{o}{\lambda}_2) = (\max(\dim(\overset{o}{\lambda}_1), \dim(\overset{o}{\lambda}_2)) - 1) \wedge (\lambda_1 \cap \lambda_2 \neq \lambda_1) \wedge (\lambda_1 \cap \lambda_2 \neq \lambda_2)$

4. Relation **recouvre** (relation ligne/ligne et surface/surface) :

   $(\lambda_1 \text{ recouvre } \lambda_2) \Leftrightarrow (\dim(\overset{o}{\lambda}_1) = \dim(\overset{o}{\lambda}_2) = \dim(\overset{o}{\lambda}_1 \cap \overset{o}{\lambda}_2)) \wedge (\lambda_1 \cap \lambda_2 \neq \lambda_1) \wedge (\lambda_1 \cap \lambda_2 \neq \lambda_2)$

5. Relation **disjoint** : $(\lambda_1 \text{ disjoint } \lambda_2) \Leftrightarrow (\lambda_1 \cap \lambda_2 = \emptyset)$

Pour ce qui est des frontières, l'opérateur renvoie une ligne pour la frontière d'une surface et les deux points extrêmes pour une ligne. Ces seules définitions suffisent pour le calcul de relations topologiques qui seront des combinaisons de ces 5 relations de base. Contrairement aux systèmes 4 et 9 intersections ou "extended-dimension", il y a très peu de notions à connaître et l'interprétation des relations est plus aisée.

On peut montrer que les 5 relations définies précédemment sont mutuellement exclusives et qu'elles sont complètes. Si on compare les différents langages, on trouve que la méthode "calculus-based" est plus expressive que le système 9-intersections ou que le système "dimension-extended". Certains cas de figures peuvent être distingués dans le premier formalisme mais pas dans les autres. La méthode "calculus-based" est en fait équivalente au système 9-intersections "dimension-extended". Le modèle "calculus-based" peut même être utilisé pour des objets plus complexes (régions avec trous, avec séparations) [Clementini&al 96].

d) Axiomatisation

Nous avons présenté ici les principaux courants de recherche sur la topologie dans le raisonnement spatial qualitatif. Pour être exhaustif, il faut encore évoquer les nombreuses tentatives d'axiomatisation de la topologie. Cette axiomatisation permet en général de retrouver des relations spatiales proches de celles fournies par RCC, mais le formalisme est une logique du premier ordre où le seul mode de raisonnement possible est la technique de la preuve, ce qui n'est pas toujours facile à mettre en œuvre.

Parmi ces tentatives on trouve celle de [Asher 95], qui parle de méréotopologie. L'axiomatisation est construite à partir du prédicat C de connexion, qui définit les différentes notions de connexion extérieure, de recouvrement et de contact faible. On ajoute en plus des aspects topologiques sous la forme d'axiomes concernant les opérations d'ouverture et de fermeture.

Jeansoulin et Galton [Jeansoulin 97], [Galton 96], ont les mêmes préoccupations d'axiomatisation. Galton part d'un prédicat P "être partie de", qui désigne une notion de contribution : un point ne peut pas contribuer à une sphère mais en revanche un petit volume le peut. Il ajoute en plus des aspects dimensionnels avec la notion d'équidimension. Jeansoulin, lui, reconstruit RCC à partir des notions de contact, de partie et d'ordre.

Davis [Davis 95] cherche un langage minimal pour un maximum d'expressivité. Son langage n'est pas une logique du premier ordre puisqu'il ne comporte pas la négation. Il est constitué d'une ontologie, de prédicats et d'une syntaxe. Les assertions sont des contraintes d'inégalité ou d'égalité. Ce langage est plus simple que tous ceux évoqués précédemment, mais le raisonnement est toujours fondé sur une recherche de preuves.

e) Relations topologiques floues

Certaines des notions topologiques classiques ont été étendues au cas de régions vagues comme on l'a vu pour les théories RCC ou 9-intersections. Mais les relations restent binaires, contrairement aux extensions utilisant la logique floue où les relations sont satisfaites avec un certain degré. Ici on s'intéresse au passage d'une relation particulière du cas précis au cas flou, et surtout aux techniques générales qui permettent ce passage. Bloch [Bloch&al 97] a en effet distingué 4 méthodes pour étendre la définition classique d'une relation à sa définition floue. Elle a appliqué ces 4 méthodes au cas de la relation d'adjacence.

Une première méthode consiste à appliquer la relation classique sur les α-coupes de l'ensemble flou et à agréger ensuite les résultats obtenus pour chaque α-coupe. Cette agrégation peut s'obtenir par des relations du type :

$R(\mu,\nu) = \sup_{\alpha\in[0,1]} \min(\alpha, R_B(\mu_\alpha,\nu_\alpha))$ ou $R(\mu,\nu) = \sup_{\alpha\in[0,1]} (\alpha R_B(\mu_\alpha,\nu_\alpha))$ où R_B est la relation classique qui renvoie une valeur numérique (dans {0,1} pour une relation booléenne, dans $\Re^+$ pour une distance…), R son extension au cas flou, μ et ν les fonctions d'appartenance des ensembles flous auxquels s'applique R et μ_α et ν_α leurs α-coupes. Le résultat obtenu par cette méthode est une relation qui s'applique sur des ensembles flous mais donne un résultat précis.

Dans certains cas on souhaite que la relation donne un résultat flou pour avoir plus de souplesse. On peut par exemple appliquer le principe d'extension :

$$R(\mu,\nu)(n) = \sup_{R_B(\mu_\alpha,\nu_\alpha)=n} \alpha$$

Une autre méthode est encore de traduire formellement la définition classique écrite sous forme d'équation logique ou analytique en remplaçant les opérateurs classiques par les opérateurs flous équivalents. On remplacera par exemple la conjonction par un ***min***, la disjonction par un ***max***… Cette méthode est utilisée dans [Sinha&al 93] pour définir l'inclusion d'ensembles flous et aussi dans [Rosenfeld 79] pour la définition des notions "connecter" et "entourer" sur des images floues. Elle conduit parfois au même résultat qu'en appliquant le principe d'extension.

Enfin, on peut définir une relation à partir de ses propriétés et des concepts qu'elle implique. Pour l'adjacence floue par exemple, Bloch [Bloch&al 97] impose les propriétés de symétrie, de consistance avec les définitions classiques (la définition floue doit donner le résultat classique sur des ensembles précis), de décroissance avec la distance entre les deux ensembles et d'invariance par certaines transformations géométriques. Ces propriétés permettent de restreindre le champ de recherche.

Il n'existe pas une méthode meilleure qu'une autre pour passer d'un concept précis à un concept flou. On choisit une méthode suivant ce que l'on souhaite obtenir, si l'on souhaite vérifier certaines propriétés, si l'on veut un résultat sous forme de nombre flou ou non et aussi en fonction de la complexité de calcul autorisée.

B) Distances

Nous allons maintenant aborder un autre aspect des descriptions spatiales, la mesure ou la comparaison de distances.

a) Le Δ-calcul

Le Δ-calcul [Zimmermann 95] est un formalisme qui permet un raisonnement qualitatif sur les distances. Il consiste à comparer des distances. Ce formalisme est fondé sur une approche cognitive de notre perception de la distance : on ne sait pas évaluer précisément une distance mais en revanche on est parfaitement capable de comparer deux distances.

Le Δ-calcul permet donc d'exprimer que "a est plus grand que b de x", ce qui s'écrit : a(>,x)b et est équivalent à a=x+b. Voici un exemple de représentation graphique de la relation a(>,x)b :

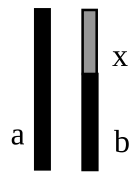


Fig. 92 : relation a(>,x)b

Le raisonnement sur de telles relations se fait en utilisant la composition de deux relations :

a(>,x)b et b(>,y)c⇒a(>,"x,y")c ou encore a(>,x)b et b(>,y)c⇒[a(>,z)c et z(>,y)x]

Ainsi s'effectue la propagation des relations entre distances. On peut avoir des disjonctions de relations, le résultat est alors la disjonction de toutes les combinaisons possibles entre les relations élémentaires. Avec les disjonctions il y a un risque d'explosion combinatoire puisque si l'on compose deux relations qui sont des disjonctions de deux éléments le résultat est la disjonction de 4 éléments. Pour limiter cette explosion, on simplifie l'écriture de a(>,x)b ou a(>,y)b en a(>,Δ)b où Δ peut prendre une valeur quelconque, et une relation inconnue s'écrit a?b.

b) Définition de distances qualitatives

La notion de distance est rarement utilisée seule, elle est souvent combinée avec des relations topologiques ou des relations de directions pour définir des positions relatives. Dans ce cadre on peut utiliser une décomposition de l'espace en distances qualitatives "près", "loin", "très loin"… Mais la définition des distances réelles correspondant à ces distances qualitatives est très subjective. Plusieurs éléments ont une influence dans cette définition, comme par exemple la distance réelle, le contexte, le facteur d'échelle… La notion "près" n'est certainement pas définie de la même façon lorsque l'on se situe à l'échelle planétaire ou à l'échelle d'un pays.

Gahegan [Gahegan 95] montre comment on peut prendre en compte ces différents facteurs pour la définition de distances qualitatives. Ceci passe par la construction intermédiaire d'une partition de l'univers des proximités, réels de [0,1]. On établit une correspondance entre les degrés de proximités et les distances qualitatives : "très près" correspond à une proximité de [0,0.2], "près" à [0.2,0.4]… La définition des distances qualitatives revient ensuite à définir la fonction de proximité.

On peut construire plusieurs fonctions de proximité suivant le considéré. La proximité globale est obtenue en combinant ces différentes fonctions. Une première fonction de proximité dépend du rapport entre la distance absolue et une distance de référence représentative du contexte (la plus grande distance possible par exemple).

La distance qualitative entre deux éléments dépend aussi de la répartition de tous les autres éléments. Soit n le nombre d'objets de la figure à analyser, on les ordonne suivant leur distance à l'objet de référence. L'objet de référence a le rang 0 et le plus éloigné le rang n. Une deuxième fonction de proximité à prendre en compte dépend du rang de l'élément.

D'autres aspects peuvent intervenir dans la définition des distances qualitatives. La proximité d'un élément peut par exemple dépendre du chemin permettant d'atteindre celui-ci. La distance devra alors être calculée le long de ce chemin. Les solutions à mettre en place pour prendre en compte ces aspects annexes sont largement liées à l'application.

c) Distances floues

La notion de distance floue peut recouvrir plusieurs concepts. On ne considère ici que des distances ayant un sens métrique et nous excluons toutes les mesures de distance ou de proximité entre ensembles flous utilisées pour comparer deux objets de même nature. On distingue encore deux types de distances métriques, une distance entre deux points à l'intérieur d'un ensemble flou et une distance entre deux ensembles flous.

i) Distance géodésique floue

Une distance entre deux points d'un même ensemble peut être vue comme une distance géodésique floue [Bloch 95], [Bloch 98]. Cela correspond à une extension de la distance géodésique classique. Il s'agit de trouver le plus court chemin entre deux points, les degrés d'appartenance étant par analogie vus comme des altitudes. Bloch définit plusieurs mesures utilisables à partir d'une liste de propriétés souhaitables. La distance géodésique peut être utilisée par exemple pour le guidage d'un robot mobile dans un environnement avec des obstacles. Le robot doit trouver le plus court chemin d'un point à un autre en évitant les obstacles, ce qui peut s'interpréter comme la distance géodésique floue entre ces deux points.

ii) Distance entre deux ensembles flous

De même que les relations topologiques, les distances floues peuvent être obtenues à partir de distances classiques en travaillant sur les α-coupes des ensembles, en appliquant le principe d'extension ou en traduisant des équations logiques sur les ensembles (cf. V.1.1A)e) p.179). Bloch a appliqué cette dernière méthode pour définir une distance à partir de la définition classique de distances en morphologie mathématique [Bloch 97], [Bloch 98].

Voici la définition classique de la distance morphologique $d_B(x,X)$ d'un point x à un ensemble X en utilisant $D^n(X)$ le dilaté de X par une boule de rayon n, $n \in \mathbb{N}$ et $n>1$ et $[D]^{-1}$ le complémentaire de D :

$$\begin{cases} d_B(x,X) = 0 \Leftrightarrow x \in X \\ d_B(x,X) = n \Leftrightarrow x \in D^n(X) \cap \left[D^{n-1}(X)\right]^{-1} \end{cases}$$

On peut traduire ces équations avec des opérateurs flous, Bloch ayant défini par ailleurs la notion de dilatation floue. On obtient alors une distance floue de fonction d'appartenance δ d'un point x à un ensemble flou X de focntion d'appartenance μ :

$$\begin{cases} \delta_{(x,\mu)}(0) = \mu(x) \\ \delta_{(x,\mu)}(n) = t\left[D_\nu^n(\mu)(x), C\left[D_\nu^{n-1}(\mu)(x)\right]\right] \end{cases}$$ où $D_\nu^n(\mu)$ est la fonction d'appartenance du dilaté de X par l'élément structurant ν de taille n, t est une t-norme et C un complément.

Pour passer ensuite à une distance entre deux ensembles flous, on applique la même procédure de fuzzification en prenant comme définition initiale une définition de la distance entre deux ensembles du type distance morphologique du point le plus proche ou distance morphologique de Hausdorff.

C) Positions relatives

Une autre information spatiale est la position des objets. Celle-ci peut être relative à un référentiel ou à un autre objet. Elle se décompose la plupart du temps en une distance et une orientation. Plusieurs modèles ont été proposés dans la littérature, que l'on peut regrouper en deux grandes catégories : les modèles qui reposent sur la définition de secteurs de l'espace et ceux qui sont fondés sur les projections suivant les axes d'un référentiel.

a) Sectorisation de l'espace

i) Secteurs coniques

- Position réduite

Hernandez [Hernandez 93] propose un premier modèle de position réduite, donnée par une information d'orientation et une information topologique. L'information topologique donne une indication de distance. Suivant qu'un objet touche ou est disjoint d'un autre objet il en est aussi plus ou moins éloigné.

On considère 4 relations topologiques "inclus", "recouvre", "touche", et "disjoint" et 8 orientations : "devant", "derrière", "gauche", "droite", "derrière à droite", "derrière à gauche", "devant à droite" et "devant à gauche". Ce sont les relations les plus courantes mais on peut prendre en compte un plus ou moins grand nombre de primitives suivant la précision souhaitée. Les relations combinées déterminent des secteurs de l'espace de forme conique. Voici la représentation symbolique de "B touche A et B derrière à gauche de A" :

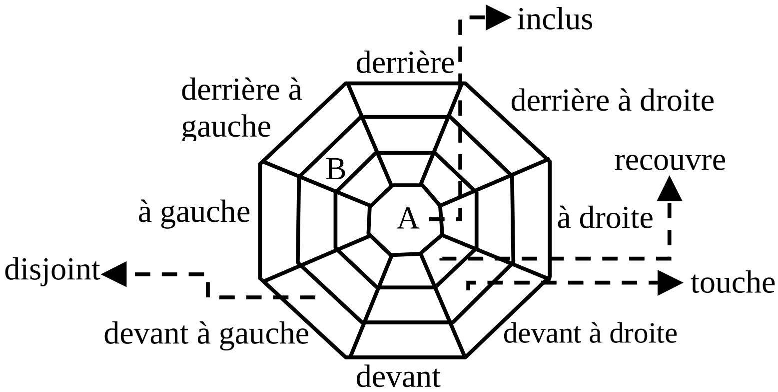


Fig. 93 : représentation symbolique de position réduite

Ici A est l'objet de référence par rapport auquel on positionne B. L'inclusion est la seule relation topologique pour laquelle on ne peut pas définir d'orientation. Dans tous les autres cas, l'orientation est donnée par rapport à un cadre de référence. Le cadre de référence peut être de 3 natures différentes : intrinsèque, extrinsèque ou déictique.

Un cadre de référence intrinsèque est défini par rapport à l'objet de référence. L'objet de référence possède en lui-même une direction de référence. Un cadre de référence extrinsèque est lui défini par un élément extérieur, par exemple par l'orientation de la pièce si l'on considère ses meubles. Enfin, le cadre de référence est dit déictique lorsqu'il est donné par le point de vue de l'observateur qui décrit la scène. La représentation symbolique précédente est valable quel que soit le type de cadre de référence employé.

L'emploi de cette représentation symbolique permet une plus grande efficacité dans le raisonnement. Un changement de point de vue par exemple revient à faire une rotation des étiquettes d'orientation pour tous les objets. Mais l'essentiel du raisonnement repose encore une fois sur la composition des contraintes afin de propager l'information. Il existe des tables de composition données pour une orientation constante ou pour une relation topologique constante. Si on traduit ces tables sur les représentations symboliques, la propagation des contraintes revient en fait à propager des jetons dans les différents secteurs.

La consistance est vérifiée par l'algorithme de type Floyd-Warshall en $O(n^3)$. On peut montrer par un contre-exemple que cet algorithme est incomplet dans le cas de telles relations spatiales. Il ne détecte pas forcément l'inconsistance. Mais l'algorithme permet néanmoins de propager l'information et de détecter la plupart des inconsistances.

- Position complète

Hernandez [Hernandez&al 95] étend cette notion de position réduite qui est insuffisante pour distinguer certaines configurations. On remplace l'information topologique par une vraie information de distance qualitative. Cette distance est donnée avec une plus ou moins grande granularité et on utilise les relations de distance (loin, près), ou (très loin, loin, relativement près, près, très près). Plus généralement on considère un ensemble de distances qualitatives (q_0, q_1,…, q_n). Le choix des distances qualitatives dépend du contexte et de la précision requise. Avec ces distances on définit comme précédemment des secteurs de l'espace en remplaçant les relations topologiques par des distances :

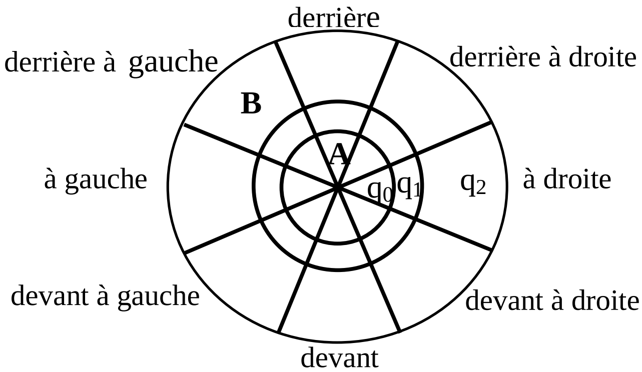


Fig. 94 : représentation symbolique de position complète

Mais il est assez difficile de raisonner avec des distances qualitatives dans le cas général, et pour simplifier on se limite à des structures particulières sur ces distances. On impose donc certaines restrictions.

Voici un schéma montrant les éléments qui interviennent dans la définition de distances qualitatives (q_0,q_1,…, q_n) :

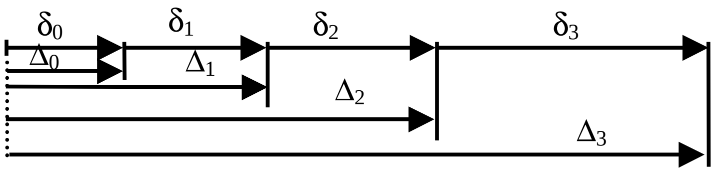


Fig. 95 : définition des distances qualitatives

Voici les restrictions que l'on peut imposer pour simplifier les calculs :

1. distances monotones : $\delta_0 \leq \delta_1 \leq \ldots \leq \delta_n$
2. $\delta_i \geq \Delta_{i-1}$
3. absorption : si $\delta_i >> \delta_j$ alors $\delta_i + \delta_j \approx \delta_j$

La restriction 2 est plus forte que la monotonie et l'absorption peut être appliquée seule ou avec les propriétés des cas 1 ou 2. Pour la propagation des contraintes, on utilise comme dans le modèle réduit des tables de composition. Mais le problème est plus complexe que dans le cas de relations topologiques car les distances, lorsqu'elles n'ont pas la même orientation, sont non additives. On ne peut que donner pour chaque cas que les bornes inférieure et supérieure de la distance résultante. Ces bornes sont calculées en tenant compte de la structure particulière des distances, et les tables sont différentes suivant les restrictions appliquées.

Si on compose les distances AB et BC, le résultat dépend aussi fortement de l'orientation de (AB) par rapport à (BC). Les 3 cas principaux concernent des directions identiques, opposées ou orthogonales. On calcule donc une table de composition pour chacune de ces 3 orientations et suivant les restrictions appliquées sur la structure des distances. On essaye ainsi d'approcher au mieux les bornes du résultat de la composition. Un algorithme donne suivant la valeur de l'orientation la ou les tables qu'il faut utiliser pour calculer les bornes inférieures et supérieures de la composition.

Ce modèle de position complète donne des résultats plus satisfaisants que le modèle de position réduite. Mais néanmoins la propagation des contraintes conduit rapidement à des résultats assez vagues où les distances et les orientations sont des disjonctions de plusieurs distances ou orientations qualitatives. L'algorithme reste incomplet, comme précédemment.

ii) Secteurs obtenus par projections

Comme on l'a vu, le modèle de Hernandez considère la position relative d'un objet par rapport à un autre, chacun étant représenté par un point. Le modèle de Zimmermann et Freksa est assez semblable puisqu'il détermine aussi des secteurs de l'espace et qu'il étudie l'opération de composition sur ces secteurs. La différence provient essentiellement du fait que l'on ne définit plus une position relative par rapport à un point mais par rapport à deux objets qui forment un vecteur. On n'utilise plus de repère.

- Définition des secteurs

Le modèle de Zimmermann et Freksa est présenté dans [Zimmermann&al 93]. A partir d'un vecteur, on définit 15 secteurs de l'espace. Ces secteurs sont des surfaces, des demi-droites ou des points. Ils correspondent à la fois à une notion de distance et d'orientation. La détermination de l'appartenance d'un point à un secteur se fait par projections sur les axes définis par le vecteur (AB). En voici une représentation :

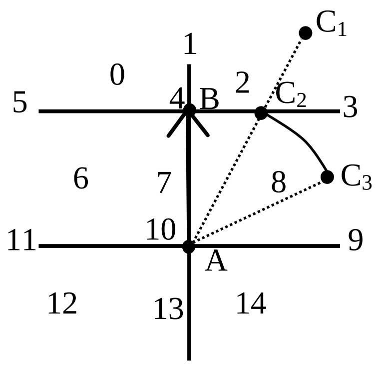


Fig. 96 : secteurs définis par projections

Les trois points C_1, C_2 et C_3 se situent respectivement dans les secteurs 2, 3 et 8. C_1 et C_2 correspondent à une même orientation par rapport à (AB) mais à deux distances de A différentes, et C_2 et C_3 correspondent à une même distance par rapport à A mais à deux orientations distinctes par rapport à (AB).

Cette représentation est tout à fait comparable à la représentation des intervalles généralisés chez Ligozat [Ligozat 97]. Un intervalle généralisé est en effet une séquence ordonné d'instants. Ces instants définissent des zones temporelles comparables aux secteurs géométriques définis dans la figure ci-dessus. La relation entre deux intervalles généralisés peut être décrite en fonction de ces zones, et les opérations sur les relations sont aussi définies par rapport à ces zones.

- Opérations

Dans ce formalisme, 4 opérations élémentaires sont définies. On écrit c:ab pour signifier que c est en relation avec le vecteur ab. Les opérations élémentaires permettent de passer d'une ou plusieurs relations initiales à d'autres relations. Ces relations sont :

1. l'inversion qui transforme c:ab en c:ba
2. le homing qui transforme c:ab en a:bc
3. le shortcut qui transforme c:ab en b:ac
4. la composition qui transforme c:ab et d:bc en d:ac

A chaque opération correspond une table qui donne la relation finale en fonction de la ou les relations initiales. On retrouve parmi ces tables la classique table de composition. On peut montrer que ces seules opérations suffisent, en les combinant, à trouver la relation qui lie n'importe quel point à n'importe quel vecteur, si l'on connaît au départ au minimum une relation faisant intervenir chaque point.

- Ajout de distances

Les secteurs de l'espace ainsi définis ne permettent pas encore de distinguer tout couple de points de l'espace. On ajoute donc des informations supplémentaires de distances [Zimmermann 93] qui permettent de définir de façon unique chaque point de l'espace. Voici les 5 distances que l'on peut utiliser, certaines pouvant être redondantes :

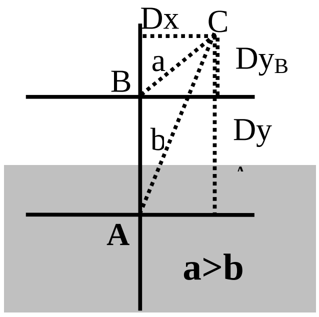


Fig. 97 : information de distance ajoutée au modèle

On peut établir des correspondances entre les distances et les régions de l'espace. La région grisée correspond par exemple à des distances telles que a>b, ce qui limite aux secteurs 6 à 14. Pour la composition des distances on utilise le Δ-calcul présenté en V.1.1B)a) p.180.

- Prise en compte de l'arrangement

Indépendamment des notions de distances, on peut améliorer le modèle en ajoutant une notion d'arrangement entre les points [Latecki&al 93]. Lorsque l'on positionne C par rapport à un vecteur (AB), on peut distinguer suivant que l'angle ABC est dans le sens trigonométrique (+) ou non (-) et s'il est obtus (o) ou aigu (a). Un observateur est a priori capable de décrire ainsi l'arrangement de 3 points sans trop de difficultés, ces notions sont assez intuitives. On peut établir des règles d'inférence pour effectuer la composition de ces relations. Un exemple de règle est :

$$ABC(+,a) \wedge CBD(+,a) \Rightarrow ABD(+,?)$$

Si on considère à la fois les angles ABC, BAC et ACB, on peut redéfinir les mêmes secteurs que précédemment et même en ajouter deux supplémentaires. Le secteur 5 correspond par exemple à ABC(-,a) et BAC(+,a) et ACB(+,o).

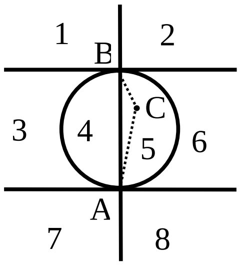


Fig. 98 : secteurs définis par les angles ABC, BAC et ACB

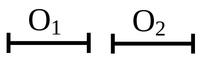
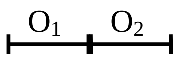
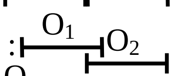
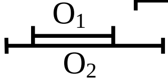
Le modèle initial de Freksa et Zimmermann est légèrement raffiné par ces notions supplémentaires d'angles puisque l'on distingue deux régions supplémentaires, mais il reste moins riche que le modèle initial auquel on ajoute les notions de distances.

b) Système de coordonnées

Une autre possibilité pour déterminer une position relative est de donner les coordonnées relatives. Ces coordonnées peuvent être cartésiennes ou cylindriques. Au lieu de comparer globalement les positions, on compare les coordonnées deux à deux. Cette comparaison peut-être aussi bien quantitative que qualitative.

Le modèle de Guesgen [Guesgen 89] est directement inspiré des relations d'Allen dans un espace à une dimension. On compare en effet des coordonnées deux à deux, donc on se place dans un espace à une dimension.

Chaque objet est caractérisé par un intervalle d'abscisses $[x_1,x_2]$ et un intervalle d'ordonnées $[y_1,y_2]$. Sur chacun des deux axes on considère des relations qualitatives semblables à celles d'Allen. Guesgen utilise en fait les quatre relations suivantes et leurs inverses :

1. O_1 "à gauche" de O_2 de symbole $<$ et d'inverse $>$: 
2. O_1 "attaché à" O_2 de symbole $\leq$ et d'inverse $\geq$: 
3. O_1 "recouvre" O_2 de symbole $\Leftarrow$ et d'inverse $\Rightarrow$: 
4. O_1 "dans" O_2 de symbole $\subset$ et d'inverse $\supset$: 

Ces relations se traduisent indifféremment sur les axes x et y à l'aide des relations <, >, et =. La relation 1 se traduit par exemple sur l'axe des abscisses par : $\forall x \in O_1, \forall x' \in O_2\ x<x'$. On peut construire un réseau où les nœuds sont les objets et les arcs sont étiquetés par les couples de relations liant ces objets suivant chaque axe.

Voici un exemple avec 2 objets :

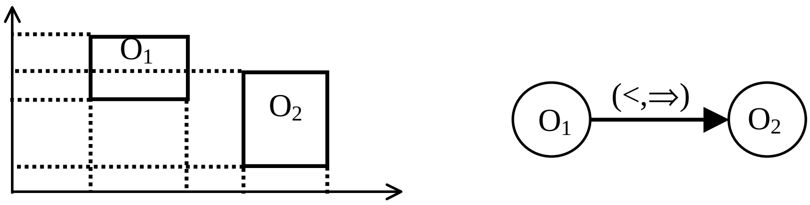


Fig. 99 : exemple de représentation sous forme de graphe

La propagation s’effectue en composant les relations suivant chaque axe. La composition de deux relations est donnée par une table de composition de même que pour les relations d'Allen. La consistance s'obtient en $O(n^3)$ par un algorithme de consistance de chemins. Une disjonction est traitée comme une disjonction de couples de relations et pas comme un couple de disjonctions pour éviter certains cas ambigus.

Enfin, pour certaines applications il est plus facile de s'exprimer en termes de coordonnées cylindriques. C'est le cas par exemple si les informations sont du type distances ou angles relatifs. Dans ce cas on considère le couple $([d_1,d_2],[\alpha_1,\alpha_2])$ des intervalles de distances et angles possibles. Le raisonnement s'applique de façon totalement identique.

Le modèle a été étendu pour prendre en compte à la fois des relations qualitatives telles que les précédentes et des relations quantitatives [Guesgen 95.2]. Le passage à de telles relations mixtes est réalisé par une représentation sous la forme d'ensembles flous. On considère en effet qu'une relation primitive du type Allen est une variable linguistique qui peut se traduire par une relation floue. De même une relation quantitative peut être traduite par une relation floue.

On peut ensuite appliquer à ces relations floues une version étendue de l'algorithme d'Allen. On remplace les opérations classiques de composition et d'intersection par leurs équivalents flous. Les résultats peuvent ensuite être retraduits sous une forme qualitative par des termes linguistiques.

c) Positions relatives de rectangles englobants

Il est parfois difficile de comparer des régions de formes complexes. Dans ces cas on se ramène souvent à un point représentatif de la région. Une autre alternative consiste à considérer les rectangles englobants de ces régions. Le rectangle englobant d'une région est le plus petit rectangle la contenant.

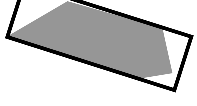


Fig. 100 : exemple de rectangle englobant

i) Parallélogramme de collision

Dans [Mukerjee&al 90] par exemple on se ramène à une position relative unidimensionnelle par l'intermédiaire d'un parallélogramme de collision que nous allons définir. Chaque objet de l'espace est associé à un rectangle englobant et à une direction qui symbolise l'avant.

Avec ces directions, on peut déterminer une direction relative d'un objet B par rapport à un objet A, par exemple en établissant quatre secteurs de l'espace. La direction relative de A par rapport à B, dir(A/B), est le secteur pointé par A dans la sectorisation relative à B. Voici un exemple :

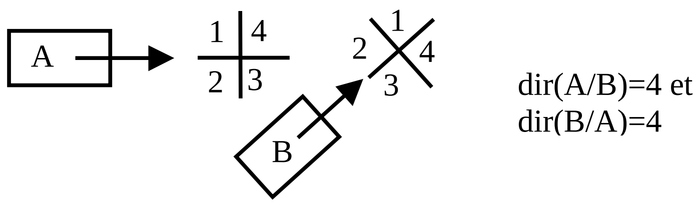


Fig. 101 : directions relatives

On définit ensuite le parallélogramme de collision (CP) obtenu par intersection des lignes directrices des rectangles englobants. Ce parallélogramme définit lui-même deux intervalles de référence par projections sur les lignes directrices de chacun des objets. L'information de position relative est donnée par la position des faces avant et arrière du rectangle par rapport à l'intervalle de référence. On utilise un couple de relations d'Allen (-,b, i, f, +) entre un point et un intervalle.

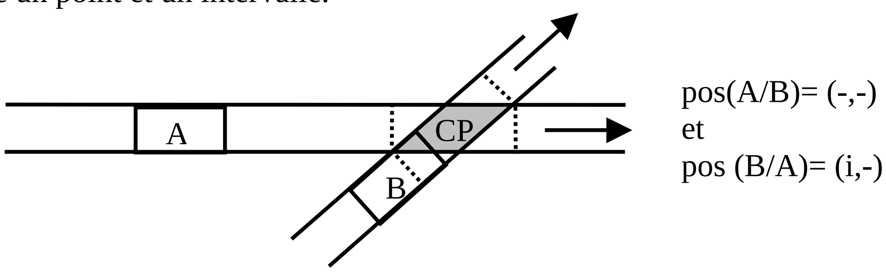


Fig. 102 : positions relatives

Le raisonnement se fait ensuite à partir de tables de composition combinant à la fois position et direction relatives.

ii) Définition de secteurs par projections

Ici on considère des rectangles englobants ayant tous la même orientation définie par rapport à un référentiel global. En prolongeant les faces d'un rectangle, on peut définir 25 secteurs différents pour définir la position d'un point par rapport à ce rectangle [Papadias 94].

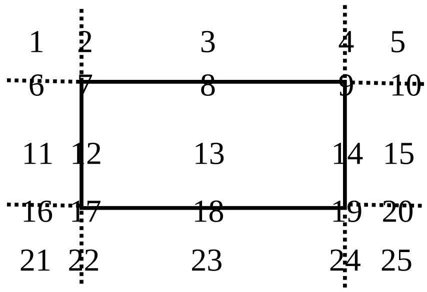


Fig. 103 : secteurs définis par un rectangle

Un rectangle est représenté par deux points, le point inférieur gauche et le point supérieur droit. La position d'un rectangle par rapport à un autre est donc donnée par la position de chacun de ses points par rapport aux secteurs définis par le rectangle de référence. Sur la figure suivante la position de B par rapport à A est (13,5) :

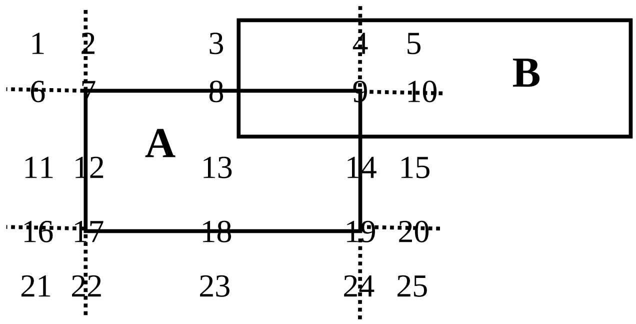


Fig. 104 : position relative de A et B

Tous les couples de secteurs ne sont pas possibles. On ne peut pas avoir par exemple (13,11) : il faudrait que le point inférieur gauche soit à droite du point supérieur droit, ce qui est impossible. Il y a en fait 169 combinaisons possibles, donc 169 relations spatiales primitives entre rectangles englobants.

On peut établir une correspondance entre ces 169 relations et les 8 relations topologiques primitives du système 9-intersections. On peut aussi définir des relations de direction entre rectangles englobants. Ces relations sont du type : strictement au nord (tous les points du rectangle sont au nord de tous les points du rectangle de référence), faiblement au nord (pour tous les points x du rectangle il existe au moins un point y du rectangle de référence tel que x au nord de y)… On peut se référer à [Papadias 94] pour les définitions exactes des relations de direction. De même il y a une correspondance entre ces 169 relations et les relations de direction.

d) Positions relatives floues

Dans les modèles qualitatifs, la précision des descriptions est fixée par le choix de la granularité des relations que l'on utilise. Les modèles qualitatifs sont un moyen de rendre compte de l'imprécision. Un autre moyen, qui appartient à la fois à la catégorie des modèles qualitatifs et quantitatifs, est l'utilisation de quantités floues. La logique floue permet de décrire aussi bien des termes linguistiques ("près de") ou des quantités vagues ("environ 10").

On retrouve les mêmes types d'approches que précédemment, étendues à des relations floues. On trouve en effet des modèles qui travaillent sur les coordonnées, des modèles qui découpent l'espace en secteurs, ces secteurs étant maintenant définis de façon floue et aussi des approches considérant les rectangles englobants. L'extension floue du modèle de Guesgen présentée en b) fait partie de cette catégorie de systèmes.

i) Système de coordonnées

Le modèle proposé par Dutta [Dutta 91.2] peut être appliqué dans des applications où le système reçoit des informations incomplètes et vagues liées à des observations et doit reconstituer une réalité cohérente.

On considère comme élément de base le point. Pour un objet, on choisit comme point de référence son barycentre ou un point particulier de l'objet. La position d'un point est fournie par une distance et une orientation. Ces informations données sous la forme d'expressions linguistiques ou de quantités vagues sont transformées en distributions de possibilités. Pour simplifier on utilise des distributions triangulaires.

Le problème des distances et des orientations est leur non-additivité. On effectue donc une projection pour transformer les coordonnées sphériques en coordonnées cartésiennes. Ces projections sont des approximations afin de garder des ensembles flous triangulaires. Un ensemble flou A est représenté par son noyau A_c et son support$[A_l,A_2]$. Voici le schéma de la projection de la position floue définie par $\{(R_c,R_1,R_2),(\theta_c,\theta_1,\theta_2)\}$:

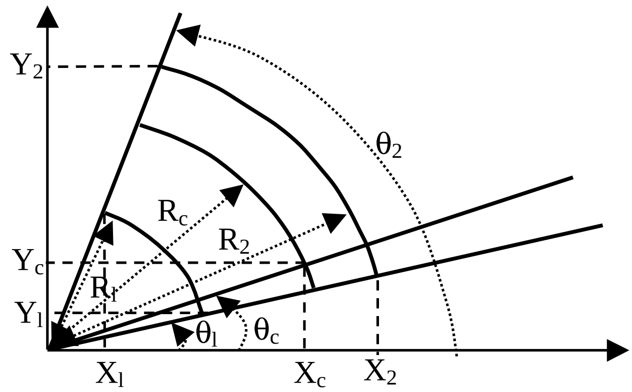


Fig. 105 : projection des distances et orientations floues

Une fois projetées sur les axes X et Y, les coordonnées peuvent se composer par addition. La composition de positions se décompose donc en deux sommes floues, une suivant X et une suivant Y. La vérification de la consistance s'obtient en appliquant l'algorithme de type Floyd-Warshall sur chacun des axes. L'algorithme est complet et permet dans le cas de la consistance de calculer les contraintes minimales en X et en Y. On peut ensuite traduire ces nouvelles contraintes en distances et orientations en effectuant la transformation inverse de la projection précédente.

ii) Rectangles englobants

Le système présenté ici a été spécialement conçu pour faire des requêtes floues sur la disposition spatiale d'objets [Cobb&al 98]. Il s'intéresse plus particulièrement aux relations topologiques et aux relations de direction. Ces relations sont construites à partir des 13 relations d'Allen. De même que dans le modèle de Guesgen [Guesgen 89], on considère que deux rectangles sont liés par un couple de relations d'Allen obtenues par projection sur les axes horizontal et vertical, chacune des projections nous ramenant à deux intervalles en dimension 1 (cf. V.1.1C)b) p.189).

Si on considère toutes les configurations possibles, on obtient 85 relations. Beaucoup de ces relations recouvrent les mêmes notions et il est possible de les regrouper. Les deux configurations suivantes correspondent par exemple à une même direction sud-ouest :

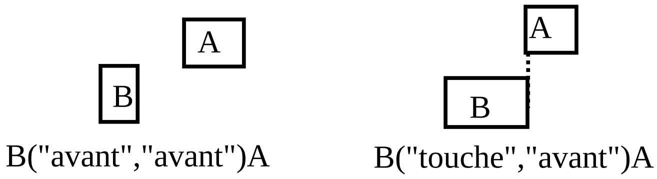


Fig. 106 : 2 configurations représentant la direction sud-ouest

On effectue deux partitions sur les couples de relations d'Allen, l'une correspondant à des notions topologiques (recouvrement, inclusion…), l'autre à des directions (nord, sud…). On ajoute en plus une notion de degré auquel la relation est vraie. Ce degré est calculé par un rapport de surfaces ou de longueurs caractéristiques. Chaque configuration permet en effet de déterminer des points, des surfaces et des segments de référence pour ce calcul. Voici un exemple de détermination des éléments de référence :

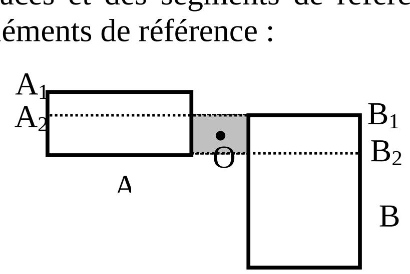


Fig. 107 : éléments de référence pour une configuration particulière

Ces éléments permettent de déterminer pour une configuration précise un degré d'appartenance à une certaine relation topologique ou de direction. En établissant un lien des valeurs numériques vers des termes linguistiques on pourra utiliser des requêtes du type : "quels sont les objets tout à fait au nord de A ?" ou "quels sont les objets qui recouvrent un peu A ?"… Ce système est utilisé pour de la description de scènes mais ne possède pas de mécanisme de propagation des contraintes englobant la notion de degré de vérification d'une relation.

iii) Sectorisation

D'autres approches utilisent plutôt une sectorisation de l'espace autour d'un objet, comme on l'a vu en V.1.1C)a) p.183. Les secteurs peuvent aussi être définis de façon floue. La définition de la sectorisation peut s'obtenir de différentes manières [Bloch 96.2], en prenant en compte un plus ou moins grand nombre de critères. Si on considère la position relative par rapport à un objet A, on peut calculer pour chaque point de l'espace son appartenance à une relation particulière avec des relations floues faisant intervenir les angles entre ce point et certains points de A par exemple. On définit ainsi un secteur flou pour chaque relation.

Lorsque l'on cherche la relation entre un objet B et un objet A plusieurs méthodes sont envisageables. On peut regarder l'appartenance d'un point de référence de B au secteur flou défini par A et la relation ou calculer cette appartenance pour plusieurs points et agréger les résultats. Une nouvelle méthode introduite dans [Bloch 96.2] permet de tenir compte aussi de la forme des objets. On compare directement l'objet flou B au secteur flou représentant la relation. La compatibilité entre ce secteur flou et B donne le degré auquel la relation est vérifiée. Cette compatibilité est donnée sous la forme d'une nécessité et d'une possibilité.

D) Formes

En dehors de la topologie, la plupart des relations spatiales considèrent qu'une région de l'espace se ramène à un point. Cette simplification est nécessaire, mais il peut aussi être intéressant de modéliser la forme d'une région. La topologie [Cohn&al 97] permet d'effectuer certaines remarques sur la forme des régions, avec dans RCC des considérations sur la connexité, les régions à trous… Mais cela n'est pas suffisant si l'on souhaite prendre en compte la forme complète d'un objet.

a) Classification qualitative

Une première approche [Clementini&al 97] tente de classer les différentes formes suivant trois axes : les axes topologique, projectif et métrique. Chaque axe permet de caractériser une forme suivant une catégorie de propriétés.

La caractérisation des formes est fondée sur la notion d'invariant. Un invariant est une propriété conservée par un certain groupe de transformations. Par exemple, la connectivité est un invariant pour les relations topologiques. Si on applique une transformation topologique sur une forme, celle-ci garde les mêmes propriétés topologiques : nombre de trous, nombre de composants connexes…

Les transformations projectives préservent elles les lignes droites et les courbes ainsi que le parallélisme. Elles permettent d'avoir des informations sur la convexité ou la concavité d'une région.

Enfin, les similarités préservent les mesures d'angles et le rapport des longueurs des segments. Deux objets A et B ont la même forme s'il existe une similarité f telle que B=f(A). Les similarités sont une source d'informations sur les aspects métriques d'une forme : élongation, axes de symétrie, taille…

On peut aussi fixer une certaine granularité dans la distinction des formes. Du point de vue des propriétés topologiques par exemple, on peut ne distinguer que les formes connexes et non connexes. Si on veut plus de précision dans la distinction, on peut considérer le nombre de composants connexes. Pour chacun des axes topologique, projectif et métrique on peut choisir un niveau de granularité.

Ainsi on peut classer deux formes suivant leurs propriétés selon ces trois axes. Ceux-ci forment un espace de configurations. Chaque type de forme correspond à un point de cet espace. Le niveau de granularité suivant chaque axe fixe la précision dans la comparaison des formes. Cette méthode peut être utilisée dans la reconnaissance d'images pour comparer un objet de l'image à un objet de référence bien connu susceptible d'apparaître dans l'image.

b) Relations géométriques floues

On peut aussi utiliser des relations géométriques floues pour mesurer la ressemblance entre une configuration géométrique et une forme particulière [Lapointe&al94]. Lapointe et Proulx utilisent des relations géométriques pour la reconnaissance de situations militaires. Le résultat de la reconnaissance dépend de la comparaison entre la situation réellement observée (positions de régiments, de chars…) et des situations théoriques.

Une situation théorique est la description d'une configuration particulière entre éléments : 3 chars formant un triangle isocèle avec une certaine orientation et une certaine taille par exemple. Dans la réalité on connaît les positions de trois chars et on veut vérifier s'ils se trouvent dans cette configuration.

La reconnaissance comporte deux aspects : d'une part la ressemblance par rapport à une forme géométrique particulière sans considération de dimension et d'autre part la taille de la configuration. On utilise des mesures de proximité pour mesurer la ressemblance par rapport à une forme géométrique. Par exemple, la mesure de proximité par rapport à un triangle isocèle en A et B vaut ($\hat{A}$ est l'angle en A) :

– $m_P(A,B,C) = 1 - \frac{|\hat{A} - \hat{B}|}{\pi}$

La taille de la figure est définie de façon imprécise par une série d'ensembles flous. On peut associer à un triangle isocèle 5 ensembles flous par exemple, un pour chacun des paramètres suivants : tailles de la base et de la hauteur et orientations de A, B et C par rapport à la direction de référence donnée par la hauteur. Le niveau de ressemblance global combine la mesure de proximité et les degrés d'appartenance des paramètres réels aux ensembles flous ainsi prédéfinis.

Tous les systèmes présentés ici sont donc utilisés a posteriori pour évaluer des formes mais ne servent pas à faire de la prédiction, il n'y a pas de propagation d'information. D'autre part ils ne traitent que le cas où l'on a des informations précises.

V.1.2 Représentations symboliques

Tous les raisonnements spatiaux présentés jusqu'ici s'appuient sur une description physique des objets. D'autres raisonnements sont fondés sur une description symbolique des objets. Les aspects numériques sont alors totalement absents mais on peut établir des relations qualitatives à partir de ces descriptions.

A) Structures hiérarchisées

a) Quadtrees

Les quadtrees [Frank 97] sont une forme de représentation symbolique hiérarchisée de l'image. Cette représentation est coûteuse du point de vue du stockage de l'information mais relativement économique pour le raisonnement. Les quadtrees reposent sur une décomposition hiérarchique de l'espace avec une granularité de plus en plus fine. On peut arrêter la description à un niveau de précision donné.

Une image est découpée en secteurs. Si un secteur ne contient que des objets de même type, il reste tel quel, sinon il est à nouveau décomposé. Une image ainsi décomposée peut se représenter sous la forme d'un arbre particulier. Voici un exemple de représentation. On considère ici une image binaire où les secteurs sont noirs (N), blancs (B) ou mixtes (M). On décrit les secteurs de gauche à droite et de haut en bas.

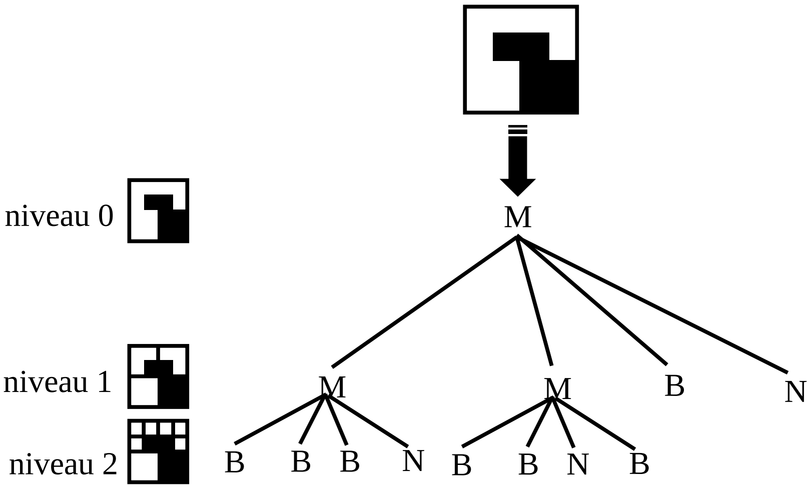


Fig. 108 : représentation d'une image par quadtrees

Cette forme de représentation peut être utilisée par exemple pour le calcul de surfaces ou pour le calcul d'intersections de régions. La surface de A est la somme des surfaces de tous les secteurs qui contiennent A. Mais cette représentation n'est pas suffisante pour décrire des relations spatiales complexes entre éléments.

b) Tableaux symboliques hiérarchisés

Les tableaux symboliques sont aussi des images symboliques hiérarchisées [Papadias&al 94]. De même que dans les quadtrees, on décompose un objet d'une image symbolique en une nouvelle image symbolique. Mais cette fois-ci le découpage se fait en tableaux de tailles variables. Chaque objet apparaît au maximum une fois par tableau, mais peut apparaître dans plusieurs tableaux. Les tableaux symboliques sont utilisés pour représenter des relations spatiales de direction.

Voici un exemple de tableau symbolique représentant l'Europe de l'ouest (we) avec ses différents pays : Irlande (ir), Grande Bretagne (br), France (fr), Portugal (po), Espagne (es), Irlande du Nord (ni), République d'Irlande (ri), Ecosse (sc), Pays de Galles (wa) et Angleterre (en). L'Irlande du Nord apparaît deux fois : en Irlande et en Grande Bretagne.

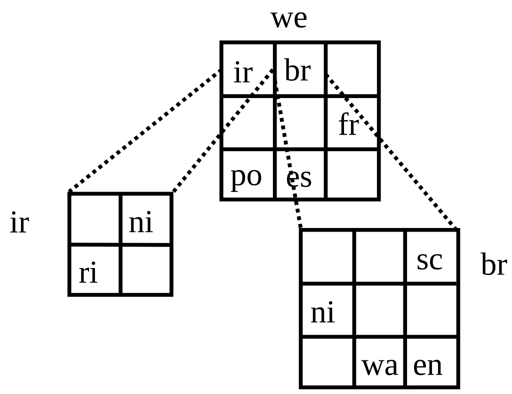


Fig. 109 : exemple de tableau symbolique

Cette représentation nous indique par exemple que l'Irlande du Nord est au nord-est de la République d'Irlande. On peut déduire pour chaque image symbolique les relations cardinales entre les objets présents dans le tableau simplement en comparant leurs numéros de ligne et de colonne. $S_{ij}(P)$ signifie que P est dans la case ligne i et colonne j. Un exemple de règle est :

$$P \text{ est au nord-ouest de } Q \Leftrightarrow \exists\, S_{ij}, \exists\, S_{kl} \,/\, \{S_{ij}(P) \text{ et } S_{kl}(Q) \text{ et } i<k \text{ et } j>l\}$$

Pour trouver la relation entre deux objets présents dans deux tableaux différents, on utilise la propagation des contraintes. Classiquement, on cherche la relation R_3 entre A et B, B étant un élément d'un objet de plus haut niveau C (B "dans" C) et C étant en relation R_1 avec A. Dans ce cas R_3 est la composition de R_1 et "dans" et elle est obtenue grâce à la table de composition.

B) 2D-string

a) Description classique

Une 2D-string est obtenue en projetant sur les axes vertical et horizontal les relations de précédence et d'égalité. Les points considérés pour établir les relationssont les points en bas à gauche des rectangles. La 2D-string se compose de deux vecteurs u et v donnant les séquences horizontale et verticale de gauche à droite et de bas en haut.

Voici un exemple de 2D-string :

u : a<b=c<d
v : d<b=a<c

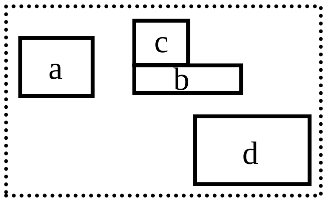


Fig. 110 : exemple de 2D-string

b) Relations de tangence

Les seules relations < et = ne permettent pas de distinguer des images un peu plus complexes. On peut leur ajouter différentes séries d'opérateurs, chacune permettant de faire ressortir un aspect de l'arrangement des objets dans l'espace. Une première extension concerne la notion de tangence : a|b signifie que a est tangent à b. On peut raffiner cette notion de tangence par les trois opérateurs décrits ci-dessous :

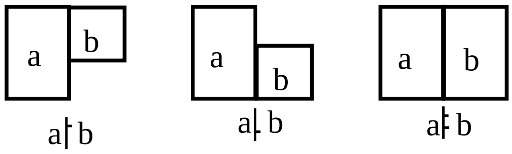


Fig. 111 : relations de tangence

c) Relations hiérarchiques

Pour l'instant ces relations ne fournissent pas d'information sur la hiérarchie des objets. On ajoute donc les 16 relations suivantes. Les quadtrees ainsi construits sont aussi appelés 2D-H-string. Ces relations sont en fait utilisées par paires :

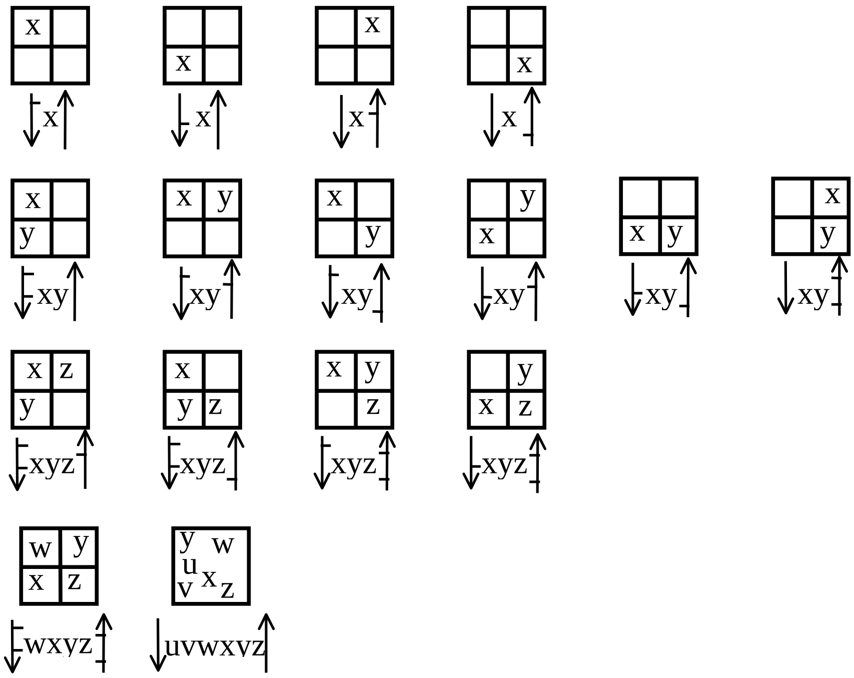
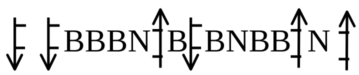


Fig. 112 : relations hiérarchiques

C'est une autre manière de représenter l'arbre de décomposition des quadtrees présentés en V.1.2A)a) p.197. L'exemple donné en V.1.2A)a) peut s'écrire avec ces relations :



d) Relations locales

Enfin, une dernière série de relations permet de tenir compte des positions relatives des extrémités des objets, ce qui dans certains cas autorise aussi certaines considérations sur les distances. Voici ces relations :

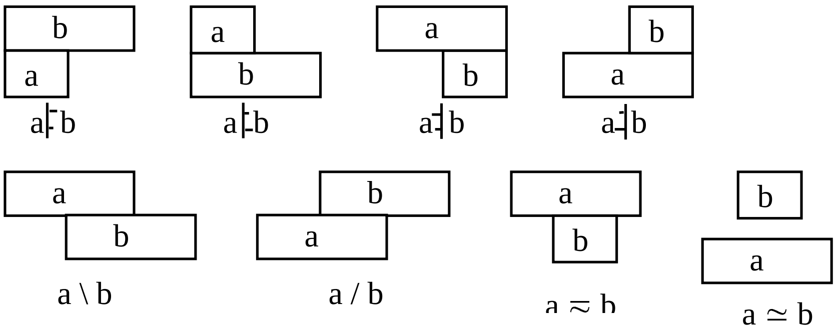


Fig. 113 : relations locales

e) 2D-string généralisées

Chaque série de relations met l'accent sur un aspect des relations spatiales, mais on ne peut pas les utiliser toutes ensemble. On ne considère donc pas tous les aspects spatiaux en une même représentation sous forme de 2D-string. C'est pourquoi [Chang&al 89] ont introduit une représentation qui subsume toutes les autres, c'est la notion de 2D-string généralisée ou 2D-G-string. Les 2D-string généralisées combinent l'intérêt d'une représentation hiérarchisée et permettent de décrire des relations spatiales plus complexes que les 2D-string classiques.

Le principe des 2D-string généralisées est le suivant : on segmente l'image horizontalement et verticalement puis pour chaque secteur horizontal (resp. vertical) on décrit les éléments présents de bas en haut (resp. de gauche à droite), l'élément e étant l'espace vide. Voici un exemple de 2D-string généralisée :

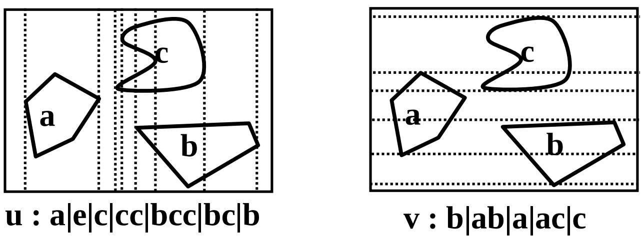


Fig. 114 : exemple de 2-D-string généralisée

A partir de cette représentation, on peut établir des règles qui concluent à des relations spatiales en fonction d'une certaine structure de la représentation. Par exemple on pourrait avoir la règle : P est au nord de Q s'il existe dans u une séquence |…Q…P…| et dans v une séquence Q|…|P. On retrouve toutes les notions exprimables dans les formalismes précédents puisque cette représentation subsume les autres, donc on pourrait établir des règles pour les relations cardinales, les relations de tangence…

f) Représentation symbolique

Une autre façon de comparer des formes d'un point de vue qualitatif est de représenter chaque forme par une description symbolique qualitative. Jungert [Jungert 93] décrit une forme par une succession d'angles convexes ou concaves, aigus, droits ou obtus et plus petits, plus grands ou égaux à l'angle précédent dans la liste. Cette description est obtenue par une technique de représentation en 2D-string construite par projection sur les axes parallèlement à une droite. Les formes considérées sont représentables par une liste de segments qui se touchent (polynômes, lignes brisées…).

On projette les points d'une forme parallèlement à une des droites constituant l'angle et parallèlement à la droite orthogonale à la première. Les points obtenus sont ordonnés sur les axes X et Y. On décrit ensuite les relations qualitatives entre les points résultant des projections. Voici un exemple de cette représentation pour un angle.

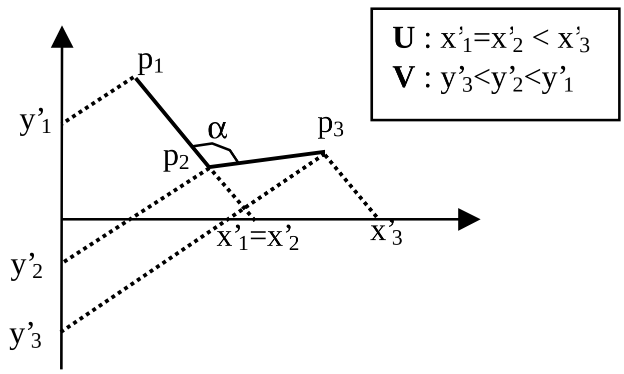


Fig. 115 : représentation symbolique d'un angle

On peut établir une correspondance entre ce type de représentation symbolique et une description qualitative de l'angle donnant sa convexité ou sa concavité et déterminant s'il est aigu, obtus ou droit. En parcourant ainsi tous les angles successifs d'une forme, on peut lui associer une description qualitative de chacun de ses angles. Pour compléter cette description, il reste à comparer la valeur de deux angles successifs. La méthode est encore fondée sur des projections représentées sur le schéma suivant :

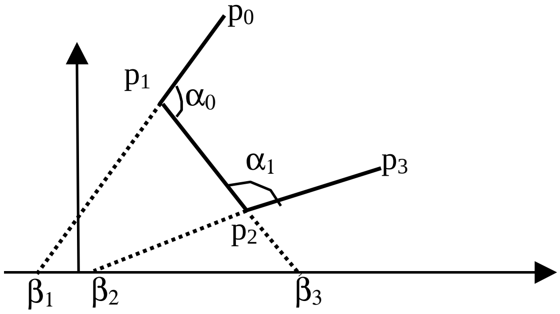


Fig. 116 : représentation de deux angles successifs

On peut là aussi établir des règles permettant de déduire une relation entre α_0 et α_1 en fonction de leurs descriptions qualitatives (natures des angles) et en fonction de β_1,β_2 et β_3. La description qualitative d'une forme se résume donc à la liste des descriptions qualitatives de chacun de ses angles, deux angles successifs étant liés par une relation qualitative de comparaison (<, > ou =). Deux formes sont considérées comme identiques si leurs descriptions qualitatives sont identiques.

V.1.3 Modèles spatiaux numériques imprécis

On trouve aussi dans la littérature des modèles qui traitent explicitement les imprécisions sous une forme numérique. Ce sont essentiellement le modèle potentiel et le modèle "haze-point". Dans le modèle potentiel on retrouve la notion de relation partiellement satisfaite comme en logique floue.

A) Modèle potentiel

Le modèle potentiel [Yamada 93] est une modélisation spatiale construite par analogie avec la physique. A chaque relation spatiale est associée une fonction d'énergie potentielle. Les valeurs des fonctions d'énergie potentielle vont de 0 à $+\infty$. Plus la valeur est faible, plus la relation est satisfaite. Lorsque l'on a plusieurs relations à prendre en compte, on superpose les fonctions d'énergie potentielle. La configuration la plus probable ou la plus satisfaisante est obtenue lorsque la fonction d'énergie potentielle globale est minimale. On se ramène donc à un problème d'optimisation. Le minimum est obtenu par la méthode du gradient.

La difficulté réside dans la construction des fonctions d'énergie potentielles. Il faut construire une fonction pour chaque primitive spatiale. Pour modéliser une relation de distance, on fait appel une fois de plus à une analogie avec la physique. Deux objets contraints par une distance relative vague sont comme deux objets reliés par un ressort. La position idéale correspond à un ressort de longueur nominale, ni comprimé ni étiré. Son énergie potentielle est alors nulle et elle grandit si on l'étire ou si on le comprime.

Soit un ressort de raideur K et de longueur L attaché en ses deux extrémités A et B distantes de d(A,B). Sa fonction d'énergie potentielle est :

$$P(A,B)=\frac{K(d(A,B)-L)^2}{2}$$

Cette fonction d'énergie potentielle modélise la relation de distance relative entre deux objets. La raideur K représente la précision de la relation. Plus K est faible, plus la relation est tolérante vis-à-vis des distances différentes de L.

Par analogie, on peut construire une énergie potentielle pour représenter une relation de direction. Soit un vecteur u contraint à être dans la direction du vecteur v et θ(u,v) l'angle entre u et v. L'énergie potentielle vaut alors :

$$P(u,v) = \frac{K}{2}\theta(u,v)^2 = \frac{K}{2}\arccos^2\left(\frac{\vec{u}.\vec{v}}{\|\vec{u}\|*\|\vec{v}\|}\right)$$

Le modèle du ressort ne permet pas de modéliser tous les types de relations. Il ne permet pas par exemple de décrire qu'un objet est situé à l'intérieur d'une région. Pour ce type de relation on utilise la notion de région interdite. Une région interdite est composée de plusieurs demi-espaces interdits. En superposant ces demi-espaces on peut construire les frontières d'une région et éliminer tous les points à l'extérieur de cette région.

En 2 dimensions la fonction d'énergie potentielle d'une région interdite est donnée par rapport à une équation de droite et elle est de la forme :

$$P(x,y) = \begin{cases} +\infty \text{ si } ax+by+c<0 \\ \quad 0 \text{ sinon} \end{cases}$$

En combinant ces différents types de fonctions d'énergie potentielle, on obtient un pouvoir d'expression riche. Pour exprimer par exemple que "A est au nord de B", on peut combiner la fonction d'énergie potentielle donnant la direction nord et la fonction interdisant la région sud. En effet, "A est au nord de B" peut être interprété par "les directions ouest et est sont à la limite acceptables" mais "la région sud est elle totalement interdite".

Un problème de contraintes spatiales peut ainsi se traduire en une fonction d'énergie potentielle globale. La solution du problème est donnée pour une valeur minimale de cette fonction. Le minimum est obtenu par la technique classique du gradient. Un problème peut être sur-contraint ou sous-contraint. Dans le premier cas le minimum a une valeur supérieure au seuil au dessus duquel on décide que le problème est inconsistant. Dans le deuxième cas on a plusieurs minima : il y a plusieurs interprétations ou plusieurs solutions au problème.

B) Modèle "haze-point"

Le modèle "haze-point" est fondé sur une logique du premier ordre représentant des points vagues [Topaloglou 94.1]. Un point x en dimension 1 est vu comme un intervalle centré sur x et de largeur 2g. g est un paramètre indiquant la précision que l'on a. En dimension 2, un point est un carré de côté 2g. On construit deux relations entre ces points. Une relation appelée "haze" notée h et une relation de précédence $<_g$. Leurs définitions sont : $h(x,y,g) \Leftrightarrow |x-y| \leq g$ et $x <_g y \Leftrightarrow y-x > g$.

On ajoute à ces définitions une série d'axiomes. La théorie ainsi obtenue en dimension 1 est complète et décidable. On étend cette théorie à la dimension 2 en construisant les points et les relations par des paires de points et de relations de dimension 1. Cette extension repose sur l'hypothèse d'une combinaison indépendante : chaque dimension peut être considérée indépendamment de l'autre. La théorie est aussi complète en dimension 2.

Un problème décrit dans ce formalisme peut être représenté par un réseau de contraintes en dimension 1 et par deux réseaux en dimensions 2 [Topaloglou 94.2]. Les nœuds représentent les points de l'espace et les arcs dénotent soit une relation de précédence, soit une relation "haze". On peut résoudre un tel problème avec les techniques habituelles des problèmes de satisfaction de contraintes. La consistance est obtenue en O(n) et le graphe minimal en $O(n^3)$. Le détail des algorithmes est présenté dans [Topaloglou 96].

Mais quelles sont les relations spatiales représentables dans ce formalisme ? Le modèle "haze-point" permet de représenter l'enveloppe spatiale de points ou de régions de l'espace partiellement indéterminés. Cette enveloppe correspond à l'imprécision g que l'on a sur la localisation exacte d'un élément de l'espace. L'extension à 2 dimensions repose sur l'hypothèse de combinaison indépendante. Cette hypothèse impose que l'espace soit orthogonal et que les éléments spatiaux soient des points ou des rectangles de côtés parallèles aux axes.

D'un point de vue topologique, on pourra représenter les huit relations primitives obtenues par le modèle RCC8 (cf. V.1.1A)a) p.173). Ces relations s'appliquent entre des régions de frontières vagues. Ces relations sont des combinaisons des relations h et <. On peut de même définir des relations de direction similaires à celles décrites dans V.1.1C)a)ii) p.186 ("à gauche", "à droite", "au-dessus", "en dessous"). A ces contraintes qualitatives s'ajoutent toutes les contraintes quantitatives exprimables sous la forme de relation d'inégalités (au sens de la théorie "haze-point" ou au sens classique) sur les abscisses ou sur les ordonnées dont voici des exemples : $X_A<5$, $Y_B=3$, $h(X_C,4,g) \Leftrightarrow |X_C-4| \leq g$, $Y_D <_g 8 \Leftrightarrow 8-Y_D>g$ …

V.1.4 Conclusion

Le choix d'une modélisation particulière doit considérer plusieurs critères : complexité, pouvoir d'expression et raisonnements possibles. On doit en effet être capable de modéliser toute l'information utile et d'en tirer les conclusions dont on a besoin tout en minimisant la complexité des calculs. Pour ce qui est des critères de complexité et de pouvoir d'expression, il faut examiner chaque modèle au cas par cas.

Mais pour les raisonnements on peut en fait distinguer trois grandes catégories à travers tous les modèles présentés : un raisonnement du type calcul de preuves fondé sur une axiomatisation, un raisonnement de type optimisation ou un raisonnement du type problème de satisfaction de contraintes. Chacun de ces raisonnements a ses avantages et ses inconvénients et permet certains types de raisonnement et pas d'autres.

Les axiomatisations (cf. V.1.1A)d) p.178) permettent un bon pouvoir d'expression. La difficulté réside dans la définition des prédicats et la création des différents axiomes : ces axiomes doivent traduire des exigences intuitives et il est difficile d'être exhaustif et d'avoir un nombre minimal d'axiomes. De plus, les démonstrations sont souvent difficiles, et il est aussi délicat de démontrer qu'un système est complet ou décidable, donc la complexité est difficile à estimer.

Les modèles qui raisonnent par optimisation sont aussi très complexes. C'est le plus souvent un algorithme du type gradient qui est utilisé. L'inconvénient de ces méthodes d'optimisation est que l'on ne sait pas au départ si un optimum existe et s'il est unique. La solution que l'on trouve est peut-être un optimum local qui cache une meilleure solution. Ces techniques ne permettent pas de trouver l'ensemble des solutions acceptables.

Le type de raisonnement le plus largement répandu et qui semble le plus efficace est celui qui utilise la composition des relations vues comme des contraintes. Il permet d'effectuer des raisonnements dans les deux sens contrairement aux raisonnements par optimisation. Ces raisonnements sont très généraux et peuvent s'adapter à de nombreuses modélisations. Il suffit d'avoir une modélisation à base de relations élémentaires et de savoir composer et intersecter ces relations. Les modèles doivent donc fournir des tables de composition comme dans les théories d'Allen, RCC et 9-intersections où encore dans les systèmes de positions relatives définies par sectorisation.

Une fois ces tables de composition obtenues, les calculs se font essentiellement par des algorithmes de propagation de contraintes. Suivant la modélisation, ces algorithmes peuvent être complets ou non et sont plus ou moins complexes. Si les algorithmes sont complets, on a tout intérêt à choisir les modélisations qui permettent ces types de raisonnements, pourvu que le pouvoir d'expression soit suffisant pour décrire le problème.

V.2 Besoins

V.2.1 Aspects à représenter

Parmi les différents aspects spatiaux qui viennent d'être présentés, nous allons voir ceux qui sont utiles et ceux qui sont indispensables pour la représentation de gabarits spatiaux. Nous commençons par nous limiter à reconnaître une configuration planaire, ce qui recouvre déjà un nombre important de situations. On pourra par la suite passer à des relations à trois dimensions en étendant le modèle, comme cela est fait dans beaucoup de systèmes.

Dans les gabarits, on considère des éléments spatiaux de deux formes : un simple point ou une zone. La définition d'une zone est en général vague et on peut l'approcher par un rectangle englobant. De plus, on peut considérer que tous les rectangles englobants ont la même orientation afin de les représenter facilement dans un repère cartésien.

Les relations prépondérantes que l'on trouve dans les gabarits sont des relations entre points. Les descriptions de gabarits comportent essentiellement des distances relatives et des orientations relatives exprimées à l'aide des points cardinaux. Ces deux paramètres peuvent être combinés en une position relative comme dans : "le point B est à 10 kilomètres au Sud-Ouest du point A". Les informations perçues par le système peuvent être de même des distances ou des orientations relatives mais on peut aussi avoir des informations de position absolue donnée dans un référentiel plan sous la forme (latitude, longitude) ou par des coordonnées métriques.

Distances et orientations peuvent être à la fois qualitatives ("loin", "à l'Est") ou numériques précises ou imprécises ("à environ 10 kilomètres", "à 5 degrés Nord, Nord-Ouest"). Les informations perçues sur le théâtre des opérations peuvent être fournies par des capteurs ou par des observateurs.

On rencontre aussi dans la description des gabarits quelques relations topologiques. La plus courante est l'appartenance d'un point à un zone. Enfin, on peut aussi faire référence à des configurations géométriques particulières entre points, faisant intervenir éventuellement la taille de la figure géométrique : "les points A, B et C forment un triangle isocèle".

V.2.2 Modèles envisageables

Un premier critère de choix pour une représentation est l'ensemble des raisonnements autorisés par celle-ci. Dans la reconnaissance nous cherchons à faire des inférences : "à partir d'une observation P(A) et d'une relation R(A,B) déduire P(B)". Ces raisonnements sont possibles dans les représentations du type logique du premier ordre ou dans toutes les représentations permettant la composition de relations. Les raisonnements à base de logique du premier ordre sont en général plus difficiles que ceux qui font appel aux techniques des CSP.

Nous excluons toutes les représentations symboliques qui ne permettent pas d'introduire des informations numériques. Nous n'avons en effet rencontré aucun modèle de 2D-string qui intègre des distances numériques par exemple. Il nous reste donc tous les modèles qualitatifs possédant une extension à des relations numériques et les modèles qui représentent spécifiquement les imprécisions numériques.

Ces derniers permettent en effet de représenter des informations qualitatives aussi bien que numériques. Le modèle "haze-point" fait l'hypothèse de combinaison indépendante, ce qui exclut toute représentation de distance. Une distance est en effet une combinaison des coordonnées cartésiennes. Le modèle potentiel permettrait lui de représenter la plupart des aspects utiles, mais les modes de raisonnement offerts ne sont pas suffisants. Avec un ensemble d'informations données on trouve une solution mais on ne trouve pas l'ensemble des solutions envisageables. Or dans le domaine de la reconnaissance, ce qui nous intéresse est l'ensemble des positions satisfaisantes pour chaque élément.

Enfin il nous reste les différents modèles qualitatifs, chacun étant centré sur la modélisation d'un aspect spatial particulier. Pour les aspects topologiques, tous les modèles que nous avons présentés font de l'inférence de nouvelles relations topologiques. Mais dans le renseignement la plupart du temps on ne cherche pas à connaître la relation topologique qui lie deux éléments. On connaît en général la relation qui lie deux éléments et on souhaite déduire de la position de l'un la position ou au moins la zone de l'autre élément. On essaiera en fait de se ramener à des relations entre points ou entre intervalles comme le fait Guesgen V.1.1C)b) p.189 pour exprimer des relations de positions.

Les relations concernant la géométrie sont très complexes à traiter. Elles font intervenir plusieurs points et éventuellement un paramètre de dimension. Dans tous les systèmes que nous avons présentés il n'y a pas d'inférence à partir de la géométrie mais simplement une vérification des contraintes ou le calcul du degré de satisfaction des contraintes dans le cas de Lapointe V.1.1D)b) p.196. Si on sait que A, B et C forment un triangle et que A et B sont connus on ne peut pas déduire la position de C. Ce problème va bien au-delà de notre périmètre de recherche car il fait intervenir des contraintes n-aires. "La distance AB est égale à la distance BC" peut être vu comme une contrainte tertiaire entre les points A, B et C ou comme une contrainte binaire d'égalité entre deux distances qui sont elles-mêmes des contraintes. Nous nous contenterons de mesurer la compatibilité entre des positions réelles et des contraintes géométriques floues exprimées par des relations floues comme chez Lapointe.

Pour ce qui est des distances, nous avons présenté deux modèles considérant la distance indépendamment de tout autre paramètre : le Δ-calcul V.1.1B)a) p.180 et les distances floues de Bloch V.1.1B)c) p.182. Le premier système est axé sur la comparaison qualitative de distances. Il est utile pour qualifier des formes géométriques en comparant les longueurs des différents côtés par exemple, mais pas pour inférer des positions ou des distances. Connaissant la distance de A à B et la distance de B à C on ne peut pas inférer la distance de A à C. De même Bloch définit des distances par des relations floues mais il n'y a pas non plus d'inférence possible. Ces relations sont simplement utilisées pour mesurer la satisfaction des contraintes.

Les modèles de positions relatives permettent d'introduire des distances. Le modèle de Hernandez V.1.1C)a) p.183 ramène toutes les distances à des distances qualitatives donc on a une perte d'information dans le cas d'une distance numérique. Freksa V.1.1C)a)ii) p.186 traite lui l'information de distance par l'intermédiaire du Δ-calcul. Le Δ-calcul fournit une comparaison qualitative des distances donc le résultat de l'inférence perd toute information numérique de distance.

Tous les modèles travaillant sur des rectangles englobants sont spécifiques à l'expression de relations entre régions de l'espace. Ils ne peuvent pas être à la base de notre modélisation car les relations prépondérantes dans les gabarits sont des relations entre points. Finalement, le modèle qui répond le mieux à nos besoins est celui de Dutta V.1.1C)d)i) p.193. La position relative floue intègre à la fois des informations qualitatives et numériques. Mais on se limite à des coordonnées cartésiennes mal adaptées pour représenter les notions de distances.

Avec une telle représentation on aura une grosse perte d'information si l'on a une description incomplète de la situation incluant des distances. Prenons le cas de l'information : la distance entre A et B est entre 10 et 15 kilomètres. En voici une représentation géométrique avec la traduction selon les coordonnées cartésiennes :

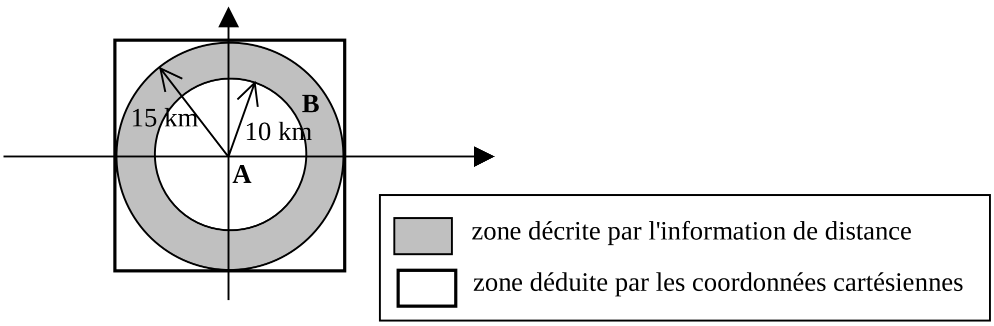


Fig. 117 : perte d'information par le passage aux coordonnées cartésiennes

Après passage aux coordonnées cartésiennes, le point d'origine (0,0) est autorisé alors qu'il ne vérifie pas du tout la contrainte de distance initiale. On a le même problème pour des directions.

Finalement, aucun des modèles présentés ne nous convient parfaitement. Nous pouvons résumer ainsi les exigences par rapport à notre modèle :

- il doit permettre de modéliser des relations qualitatives ou numériques, les relations numériques pouvant être imprécises
- il doit pouvoir intégrer des informations spatiales de position, de distance et de direction
- il doit permettre de propager les relations et de déduire les positions possibles des objets

Les relations floues semblent les mieux adaptées pour représenter des relations qualitatives et numériques éventuellement imprécises. Mais il n'existe aucun modèle utilisant de telles relations et qui mêle des informations de différentes natures. Nous allons donc introduire un nouveau modèle utilisant des relations floues et permettant de manipuler différents langages.

V.3 Modèle

V.3.1 Représentation avec plusieurs langages

La difficulté de représentation qui ressort de l'analyse précédente est qu'il faut modéliser des aspects spatiaux qui appartiennent à différents formalismes. D'un côté on souhaite décrire des zones ou des points de l'espace par leurs coordonnées cartésiennes et de l'autre il faut traiter des informations de distance et de direction qui s'expriment de façon optimale avec les coordonnées cylindriques. On doit donc manipuler deux langages simultanément.

Ce problème a déjà été abordé dans le contexte des contraintes temporelles. Dans [Kautz 91.1] on modélise effectivement à la fois des contraintes temporelles numériques entre points et des contraintes symboliques entre intervalles. Toutes les contraintes symboliques entre intervalles ne peuvent pas se traduire par des contraintes numériques entre points et inversement on ne peut pas décrire une contrainte numérique entre points dans l'algèbre d'intervalles. La solution adoptée est donc de décrire la totalité des contraintes par deux graphes distincts, un TCSP et un graphe d'Allen.

La propagation s'effectue séparément dans chaque graphe et ensuite on fait une traduction optimale des contraintes d'un formalisme dans l'autre. Même si une contrainte de l'algèbre d'intervalles ne peut pas se traduire par une contrainte numérique, elle peut inférer certaines contraintes numériques. Réciproquement une contrainte numérique peut avoir des conséquences sur les intervalles. L'algorithme de propagation global par 3-consistance n'est pas complet mais donne une bonne approximation des graphes minimaux.

V.3.2 Représentation par plusieurs graphes

Par analogie avec ce qui vient d'être décrit, nous représentons les descriptions spatiales par trois graphes distincts. Le premier graphe permettra de décrire les positions, le deuxième les distances et le troisième les directions. Mais il faut choisir un élément spatial de base pour la représentation. Nous avons vu que l'on pouvait avoir à représenter soit des points soit des régions. On choisit le point comme élément de base et une région sera représentée par plusieurs points. On choisit un référentiel extrinsèque avec un point origine arbitraire appartenant au théâtre des opérations. L'axe des abscisses peut pointer vers l'Est par exemple et l'axe des ordonnées vers le Nord. Le référentiel est le même pour les trois graphes représentant les positions, les distances et les directions. Un point P est donc désigné par ses coordonnées cartésiennes (x,y) dans ce référentiel.

Les régions R peuvent être approchées par des rectangles englobants orientés selon le repère. Elles sont donc définies par un intervalle d'abscisses $[x_1,x_2]$ et un intervalle d'ordonnées $[y_1,y_2]$. On peut les représenter par leur point de référence, centre de gravité (x_G,y_G) et par deux points frontières : le point inférieur gauche (x_1,y_1) et le point supérieur droit (x_2,y_2). Deux points seraient suffisant pour représenter un zone mais on conserve ces trois points car certaines relations s'expriment par rapport au centre de gravité et d'autres par rapport aux points frontière. La représentation des éléments spatiaux est illustrée sur la figure suivante :

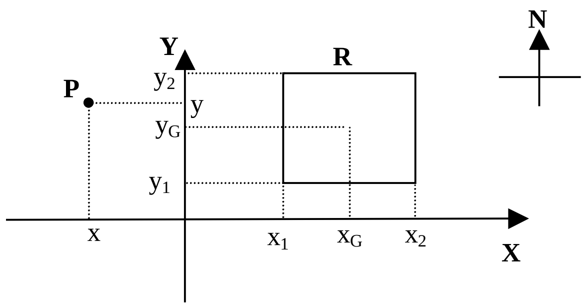


Fig. 118 : représentation des éléments spatiaux de base

Le graphe de positions va permettre d’exprimer les positions absolues et les positions relatives entre points suivant les coordonnées cartésiennes. Les nœuds du graphe sont des points et les contraintes sont des vecteurs en coordonnées cartésiennes. Dans le graphe de distances les nœuds sont aussi des points et les contraintes des distances. Le graphe de directions est similaire en prenant des directions en radians pour les contraintes. La direction entre deux points A et B est l'angle entre un axe passant par A et parallèle à X et la droite (AB). Cet angle est dans $[0,2\pi[$ et est donné suivant le sens trigonométrique. Les contraintes sont toutes numériques et sont données par des nombres ou des vecteurs flous. Un vecteur flou est un couple de nombres flous.

Voici un exemple de description de scène avec sa représentation géométrique et sa modélisation par les trois graphes de contraintes avant propagation. A est un point de coordonnées (5km,10km). B est à 10 km à l'Est et 4 km au Sud de A ce qui se traduit par : le vecteur $\vec{AB}$ vaut (10 km, -4 km). C est situé à 10 km de A, C est à l'Ouest de l'origine O (direction dans [π/2,3π/2]) et à 210° (7π/6) au Sud, Sud-Ouest par rapport à B. Les positions relatives, distances ou orientations relatives encore inconnues sont représentées par ?. Nous expliquons par la suite comment les calculer par propagation (cf. V.5 p.219).

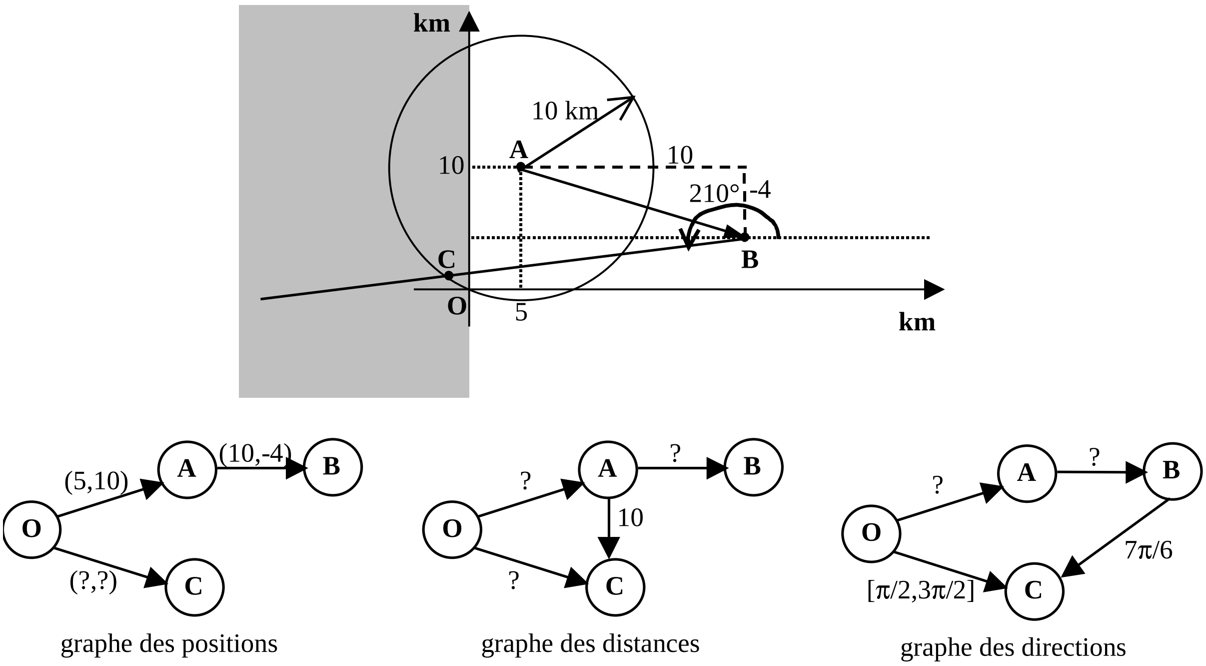


Fig. 119 : traduction d'une configuration spatiale par des graphes

On peut autoriser des contraintes disjonctives. Pour les contraintes de positions, l'expression d'une disjonction par "(x_1,y_1) ou (x_2,y_2)" est plus restrictive que "(x_1 ou x_2, y_1 ou y_2)". La première expression n'autorise pas (x_1,y_2) alors que la deuxième l'autorise. Pour éviter toute ambiguïté on utilise toujours la première forme de disjonction : on autorise des disjonctions de vecteurs où chaque vecteur comporte des coordonnées non disjonctives. La disjonction doit donc rester explicite. Pour une distance ou une direction floues, on autorise aussi les disjonctions. On peut exprimer ces disjonctions de façon explicite par un "ou" ou les représenter par un nombre flou non convexe. Cette deuxième possibilité implique une modélisation des ensembles par des fonctions affines par morceaux.

V.4 Pouvoir d'expression

Le langage de représentation des contraintes spatiales étant fixé, nous allons voir quel est son pouvoir d'expression.

V.4.1 Représentation d'une zone

Il suffit de deux points caractéristiques pour définir une zone approchée par un rectangle englobant comme on l'a vu précédemment. Les coins inférieurs gauche et supérieur droite sont en effet suffisants pour déterminer le rectangle englobant. Mais la plupart des relations font intervenir un point de référence du rectangle qui est son centre de gravité. Nous utiliserons donc les trois points Z_1, Z_2 et Z_G pour représenter une zone. Si on ne sait rien sur la taille de la zone, la seule connaissance que l'on a est que Z_1 est en bas à gauche de Z_G et Z_2 lui est en haut à droite de Z_G. On considère un rectangle englobant au sens strict avec une largeur et une hauteur non nulle (ce ne peut être ni un point ni une droite). Une zone est donc représentée par le graphe de positions suivant :

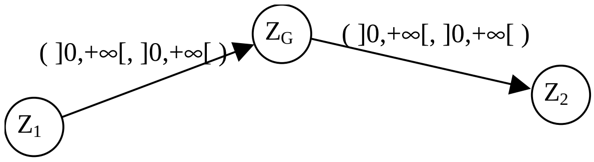


Fig. 120 : graphe de positions d'une zone

On peut aussi avoir une information sur la taille de la zone par l'intermédiaire de sa diagonale. On considérera alors la longueur de la demi diagonale d_Z qui s'intègre comme suit dans le graphe de distance :

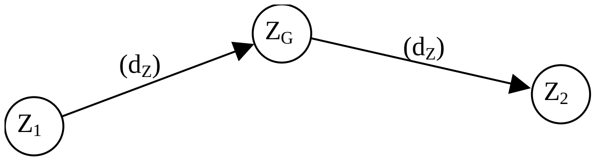


Fig. 121 : graphe de distances d'une zone connaissant sa diagonale

L'information sur la taille de la zone peut aussi être donnée sous une autre forme. On peut en effet connaître la largeur et / ou la hauteur de la zone. La largeur est entendue au sens dimension suivant l'axe X et la hauteur suivant Y. Les informations de largeur et hauteur sont à intégrer dans le graphe de positions et sont plus fortes que les contraintes initiales de la Fig. 120. On considère en fait les demi largeur et hauteur l_Z et h_Z et le graphe de positions devient :

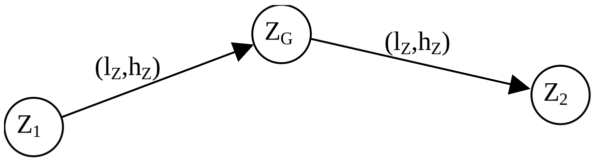


Fig. 122 : graphe de positions d'une zone connaissant sa largeur et sa hauteur

Mais tous ces graphes ne permettent pas de représenter une information importante qui est que Z_G est le centre de gravité du rectangle englobant. Il faut en fait vérifier une contrainte supplémentaire pour chacun des graphes. Ces contraintes supplémentaires doivent être exprimées de façon explicite car elles ne peuvent pas être décrites par des contraintes binaires entre points. Ces contraintes que l'on appellera ***contraintes de zone*** s'expriment de la façon suivante : pour les trois graphes de positions, de distances et de directions la contrainte entre Z_1 et Z_G est égale à la contrainte entre Z_2 et Z_G. Nous verrons dans les mécanismes de propagation comment tenir compte de ces contraintes.

V.4.2 Positions

Le graphe de positions permet d'exprimer aussi bien des positions absolues dans un référentiel fixe et des positions relatives. Les positions absolues peuvent être fournies par des cartes ou alors par des GPS (Global Positionning System) ou autres capteurs de position. Ces informations peuvent toujours être traduites sous la forme de coordonnées cartésiennes. Ces positions peuvent être imprécises. L'imprécision s'exprime par des termes comme "environ" (pour une lecture rapide de carte par exemple) ou par les caractéristiques des capteurs qui fournissent la donnée. Cette imprécision peut toujours être décrite par des ensembles flous. Les ensembles flous sont construits de façon similaire à ce qui est fait pour les contraintes temporelles en II.1.4C) p.28.

Les positions relatives sont de la même forme que les positions absolues, simplement on change le point origine du référentiel. On pourra représenter une contrainte exprimée sous la forme : "le point B est à 10 km à l'Ouest et 2 km au Nord de A". Ceci se traduit par une contrainte (-10 km, 2 km) entre A et B. La position d'une région se ramène à la position de son centre de gravité.

V.4.3 Distances

On peut aussi exprimer très simplement grâce au graphe de distances une contrainte de distance entre deux points. La distance est un nombre flou et on peut ainsi exprimer des distances numériques imprécises ("à environ 300 mètres") et des distances qualitatives ("loin"). Pour des régions on ne considère que des distances par rapport à leur centre de gravité.

La définition des distances qualitatives dépend de l'échelle de la scène : elle est fixée par la taille de la région où se déroule le théâtre des opérations. On peut donc utiliser une distance de référence pour construire les distances qualitatives de même que le fait Gahegan (cf V.1.1B)b) p.181). Cette distance de référence peut être par exemple la plus grande distance entre deux points sur le théâtre des opérations. On ne peut pas prendre en compte le nombre et la densité des éléments car la scène est connue de façon incomplète.

V.4.4 Directions

De même les directions sont représentées par des nombres flous, ce qui permet d'exprimer des directions numériques précises ou vagues ainsi que des directions qualitatives. Les directions qualitatives sont fournies par l'intermédiaire des points cardinaux. La direction Nord peut par exemple être représentée par l'ensemble flou suivant :

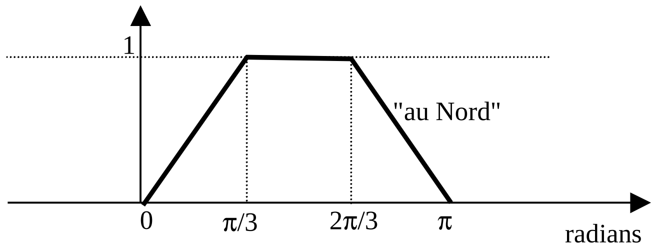


Fig. 123 : ensemble flou "au Nord"

Les directions peuvent aussi être combinées à des distances : "à 10 km au Nord-Est". Cette expression se décompose en une contrainte du graphe de distances "10 km" et une contrainte du graphe de directions "au Nord-Est".

Les contraintes de direction par rapport à une zone sont uniquement qualitatives. Cela n'a pas beaucoup de sens en effet de dire qu'un point est à une direction de 30° par rapport à une zone. En revanche on emploie couramment des directions qualitatives exprimées par les points cardinaux. Il faut distinguer le cas de la direction d'un point par rapport à une zone et d'une zone par rapport à une autre zone.

Pour ce qui est des directions qualitatives entre un point et une zone nous distinguerons 8 directions : les quatre directions élémentaires Nord, Sud, Est, Ouest et les quatre directions composées Nord-Ouest, Nord-Est, Sud-Ouest et Sud-Est. Les quatre directions élémentaires seront modélisées de la même façon. Nous choisissons la direction Nord pour détailler la construction des contraintes. Voici sur un schéma de la représentation géométrique de la relation "au Nord" pour une région R :

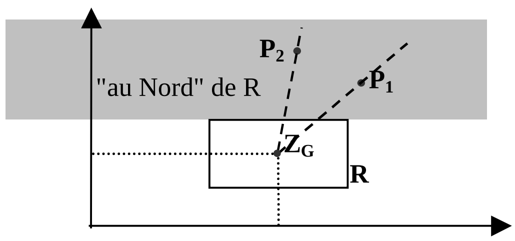


Fig. 124 : relation "au Nord" pour une région

Sur cette figure P_2 vérifie mieux la contrainte "au Nord" de R que P_1. La direction Nord délimite tout d'abord toute la région au-dessus du rectangle englobant étant donnée l'orientation de notre référentiel. Cette région est définie par l'ensemble des points d'ordonnée supérieure à y_2. Cette région se traduit pour un point P sur le graphe de positions par :

$$Z_2 \xrightarrow{(\,]-\infty,+\infty[,\ [0,+\infty[\,)} P$$

Fig. 125 : graphe de positions pour P "au Nord" de la région R

On peut assouplir cette contrainte en considérant que la région "au Nord" de R est floue. Pour cela il suffit de remplacer $[0,+\infty[$ dans la contrainte par un ensemble flou représentant le terme "positif". Il en est de même pour tous les intervalles présentés dans la suite sous cette forme.

A l'intérieur de cette région, les points peuvent vérifier plus ou moins la relation "au Nord de". La satisfaction de la contrainte dépend aussi de l'orientation du point par rapport au centre de gravité du rectangle. Plus cette orientation est proche de $\pi/2$, plus la contrainte Nord est vérifiée. L'orientation doit être dans tous les cas comprise entre 0 et π. Cette contrainte de direction correspond à la contrainte de direction Nord entre Z_G et le point considéré. Comme on l'a vu cette contrainte est un ensemble flou D_N représenté sur la figure Fig. 123. On a donc le graphe de directions suivant :

$$Z_G \xrightarrow{D_N} P$$

Fig. 126 : graphe de directions pour P "au Nord" de la région R

Pour les autres directions élémentaires la construction des contraintes est similaire. Pour une direction composée comme le "Nord-Ouest", la région acceptable est d'abord l'intersection des régions au Nord et à l'Est de R sans considérer la direction par rapport à Z_G. Cette fois-ci effectivement la satisfaction de la contrainte ne dépend plus de la direction par rapport à Z_G mais par rapport au coin supérieur gauche Z_3 de R. Ceci est illustré sur la figure suivante :

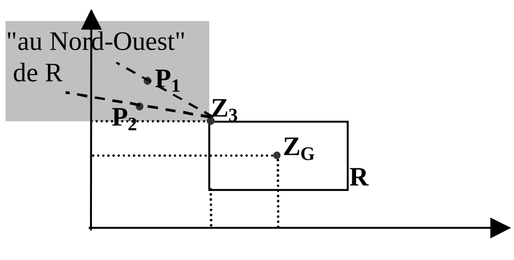


Fig. 127 : relation "au Nord-Ouest" pour une région

Sur cette figure P_1 satisfait mieux la contrainte que P_2. Plus la direction par rapport à Z_3 est proche de $3\pi/4$, plus la contrainte est satisfaite. Mais on ne souhaite pas introduire ce point Z_3 dans la représentation, ce qui augmenterait le nombre de contraintes à propager pour peu d'expressivité supplémentaire. On se contente de décrire les directions composées par des contraintes sur les coordonnées cartésiennes. Pour le Nord-Ouest, finalement on introduit des contraintes uniquement dans le graphe de positions qui devient :

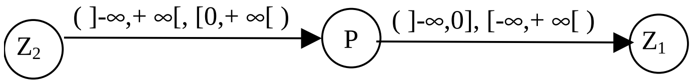


Fig. 128 : graphe de positions pour P "au Nord-Est" de la région R

On ne représente pas toutes les contraintes de régions présentées en V.4.1 p.214 pour simplifier le graphe. On peut aussi vouloir exprimer le même type de relations C entre une région R_1 et une autre région R_2. On se ramène en fait à une relation entre des points et une région. Les deux points caractéristiques extrêmes Z_1 et Z_2 de la R_1 doivent tous les deux vérifier la contrainte C avec R_2.

V.4.5 Topologie

Enfin, il peut être utile de modéliser certaines relations topologiques. La relation topologique la plus courante est l'appartenance d'un point à une région. "P à l'intérieur de R" se traduira simplement par les contraintes de position suivantes :

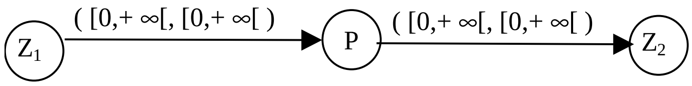


Fig. 129 : graphe de positions pour P "à l'intérieur" de la région R

Les contraintes peuvent une fois de plus être assouplies en remplaçant les intervalles par l'ensemble flou "positif".

En revanche on ne pourra pas exprimer qu'un point est à l'extérieur d'une région. "P à l'extérieur de R" est en effet équivalent à P à gauche de Z_1 ou en dessous de Z_1 ou au dessus de Z_2 ou à droite de Z_2. Il faudrait une disjonction de graphes de positions pour représenter cette relation, ce qui serait très lourd à gérer pour la propagation. On exclut donc cette relation. On pourra prendre en compte cette information uniquement si on est capable de préciser de quel côté de la région il se trouve. On se ramène alors aux contraintes qualitatives de direction précédentes.

De même on pourra exprimer qu'une région R_1 est dans une région R_2 mais pas qu'elle est à l'extérieur de R_2. "R_1 est dans R_2" se traduit par "les points caractéristiques Z_1 et Z_2 de R_1 sont dans R_2".

Enfin on peut reprendre aussi toutes les contraintes symboliques entre régions introduites par Guesgen V.1.1C)b) p.189. Ces contraintes se ramènent en fait à des primitives d'Allen entre intervalles suivant chacun des axes. Chaque primitive d'Allen peut se ramener à des contraintes entre points. Reprenons l'exemple de relation O_1 (<,⇒) O_2 rappelé sur la figure ci-dessous. Soient U_1 et U_2 les points caractéristiques de O_1 et V_1 et V_2 ceux de O_2, voici le graphe de positions pour cette relation :

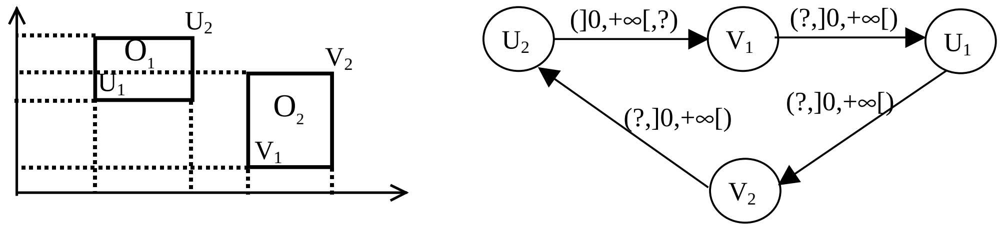


Fig. 130 : graphe de positions de la relation O_1 (<,⇒) O_2

V.5 Raisonnement

La représentation se fait donc par trois graphes distincts. Pour la propagation il faut propager les contraintes dans chacun de ces graphes et ensuite intégrer les contraintes induites par chacun des graphes sur les autres graphes.

V.5.1 Filtrage dans chaque graphe

Pour la propagation dans chaque graphe, nous utilisons un algorithme de propagation par consistance de chemins du type PC1 de même que pour le graphe temporel. Mais il faut donc préciser pour chacun des graphes comment se font les opérations élémentaires. Ces opérations sont l'inverse, l'intersection et la composition. Ces opérations servent à réaliser l'étape principale de la propagation. Mais on a vu qu'il fallait aussi ajouter certaines contraintes explicites sur les régions, contraintes qui ne peuvent être exprimées directement dans les graphes. Le traitement de ces contraintes est similaire à celui des contraintes de cohérence entre les activités mères / filles pour le graphe temporel.

A) Graphe de positions

Pour le graphe de positions une contrainte C_{ij} est de la forme (x_{ij},y_{ij}) où x_{ij} et y_{ij} sont des nombres flous éventuellement disjonctifs. Pour ce graphe les opérations élémentaires sur les vecteurs se ramènent aux opérations élémentaires sur chacun des composants du vecteur.

Les opérations élémentaires sont donc :

- inverse : l'inverse de C_{ij} est $C_{ji}=(-x_{ij},-y_{ij})$.
- intersection : l'intersection de C_{ij} et C_{kl} est : $C_{ij}\cap C_{kl}=(x_{ij}\cap x_{kl},y_{ij}\cap y_{kl})$ où $\cap$ est l'intersection de nombres flous.
- composition : la composition de C_{ij} et C_{kl} est : $C_{ij}\circ C_{kl}=(x_{ij}\oplus x_{kl},y_{ij}\oplus y_{kl})$ où $\oplus$ est la somme floue.

Il faut aussi vérifier les contraintes de zone. Ce sont toutes les contraintes inférées par le fait que pour chaque zone le point Z_G de coordonnées (x_G,y_G) est le milieu de Z_1 (x_1,y_1) et Z_2 (x_2,y_2). En terme de vecteurs ceci s'écrit aussi $\overrightarrow{Z_1Z_G}=\overrightarrow{Z_GZ_2}$. On a donc deux contraintes à appliquer (C(A,B) est la contrainte entre A et B) :

- $C(Z_1,Z_G)=C(Z_1,Z_G)\cap C(Z_G,Z_2)$
- $C(Z_G,Z_2)=C(Z_1,Z_G)\cap C(Z_G,Z_2)$

B) Graphe de distances

Pour le graphe de distances une contrainte $C_{ij}=C(i,j)$ est de la forme (d_{ij}) où d_{ij} est un nombre flou éventuellement disjonctif. Nous adoptons une interprétation géométrique des contraintes afin de déterminer les opérations élémentaires. Une contrainte de distance entre A et B correspond en fait à la norme du vecteur $\overrightarrow{AB}$. Les opérations élémentaires sont donc :

- inverse : l'inverse de C(A,B) est C(B,A) qui est la norme du vecteur $\overrightarrow{BA}$, égale à la norme du vecteur $\overrightarrow{AB}$ donc l'inverse de C_{ij} est $C_{ji}=d_{ij}$.
- intersection : l'intersection de deux vecteurs est vide si les deux vecteurs ne sont pas égaux, et vaut le vecteur lui-même sinon. Pour la norme on obtient donc l'intersection des normes : $C_{ij}\cap C_{kl}=(d_{ij}\cap d_{kl})$, toujours avec l'intersection floue.

– composition : la composition de deux vecteurs $\overrightarrow{AB}$ et $\overrightarrow{BC}$ correspond à la somme vectorielle qui est le vecteur $\overrightarrow{AC}$. Nous voyons sur la figure suivante ce que cela donne en termes de distances :

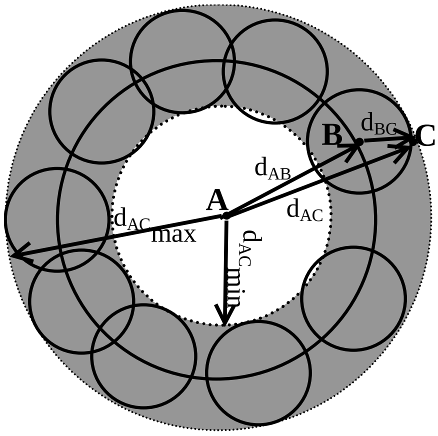


Fig. 131 : composition de distances

Sur cette figure le point C peut se trouver n'importe où dans la région grisée. La distance entre A et C est comprise entre d_{AC} min et d_{AC} max, qui correspondent respectivement à d_{AB}-d_{BC} et d_{AB}+d_{BC}. Ceci pour le cas où $d_{AB}>d_{BC}$. Pour le cas $d_{AB}<d_{BC}$ on peut effectuer la composition de BC avec AC puisque la somme de vecteurs est commutative. D'après la figure précédente et en remplaçant AB par BC et BC par AB, la distance entre C et A est comprise entre d_{BC}-d_{AB} et d_{BC}+d_{AB}. Finalement on peut synthétiser ce résultat en :

– composition : la composition de C_{ij} et C_{kl} est : $C_{ij} \circ C_{kl}=[| d_{ij}-d_{kl} |_f, d_{ij} \oplus d_{kl}]$, $| \ |_f$ étant la valeur absolue floue.

Nous avons déjà vu comment se calculait la somme et la différence floues. Il faut aussi effectuer la valeur absolue d'un nombre flou pour composer des distances. Nous verrons le calcul de la valeur absolue ainsi que d'autres fonctions en V.5.2C) p.226.

Pour la vérification des contraintes de zone, on a toujours $\overrightarrow{Z_1 Z_G} = \overrightarrow{Z_G Z_2}$,ce qui s'exprime de la même façon que pour les positions par :

– $C(Z_1,Z_G)=C(Z_1,Z_G)\cap C(Z_G,Z_2)$

– $C(Z_G,Z_2)=C(Z_1,Z_G)\cap C(Z_G,Z_2)$

C) Graphe de directions

Pour le graphe de directions une contrainte C_{ij}=C(i,j) est de la forme (θ_{ij}) où θ_{ij} est un nombre flou éventuellement disjonctif. Nous adoptons une interprétation géométrique des contraintes afin de déterminer les opérations élémentaires. Une contrainte de direction entre A et B correspond en fait à la direction du vecteur $\overrightarrow{AB}$. Les opérations élémentaires sont donc :

– inverse : l'inverse de C_{ij} est $C_{ji}=\theta_{ij}\oplus\pi[2\pi]$.

– intersection : l'intersection de deux vecteurs est vide si les deux vecteurs ne sont pas égaux et vaut le vecteur lui-même sinon. Pour la direction on obtient donc l'intersection des directions : $C_{ij}\cap C_{kl}=(\theta_{ij}\cap\theta_{kl})$, toujours avec l'intersection floue.

– composition : la composition de deux vecteurs $\vec{AB}$ et $\vec{BC}$ correspond à la somme vectorielle qui est le vecteur $\vec{AC}$.

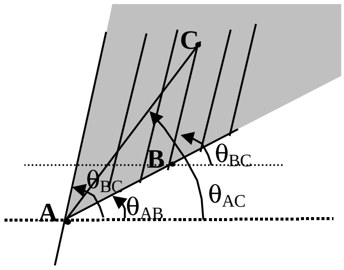


Fig. 132 : composition de directions

Sur cette figure le point C peut se trouver n'importe où dans la région grisée. C'est la cas où θ_{AB} et θ_{BC} sont dans le même demi-plan au-dessus ou en dessous de l'axe des abscisses. La direction entre A et C est comprise entre θ_{AB} et θ_{BC}. Si $\theta_{AB}> \theta_{BC}$ alors la composition des deux est dans $[\theta_{BC}, \theta_{AB}]$. Si les deux angles ne sont pas dans le même demi plan on obtient le secteur compris entre l'angle dans le demi plan inférieur et celui dans le demi plan supérieur. On peut exprimer ainsi la composition :

– composition : la composition de C_{ij} et C_{kl} est :

$C_{ij}\circ C_{kl}=[\min(\theta_{ij}, \theta_{kl}),\max(\theta_{ij}, \theta_{kl})]$ si les supports de θ_{ij} et θ_{kl} sont tous les deux dans $[0,\pi]$ ou dans $[\pi,2\pi]$.

$C_{ij}\circ C_{kl}=[\theta_{kl},2\pi]\cup[0,\theta_{ij}]$ si le support de θ_{ij} est dans $[0,\pi]$ et celui de θ_{kl} dans $[\pi,2\pi]$.

$C_{ij}\circ C_{kl}=[\theta_{ij},2\pi]\cup[0,\theta_{kl}]$ si le support de θ_{ij} est dans $[\pi,2\pi]$ et celui de θ_{kl} dans $[0,\pi]$.

Les opérateurs sont les opérateurs flous déjà définis pour les contraintes temporelles, sauf pour l'opérateur modulo qui est défini en V.5.2C) p.226.

Pour la vérification des contraintes de zone, on a les mêmes formules que précédemment.

V.5.2 Contraintes induites d'un langage à un autre

Nous considérons un espace à 2 dimensions, donc le degré de liberté pour définir un point est de deux. Il suffit de deux paramètres pour déterminer un point, ses coordonnées cartésiennes ou ses coordonnées cylindriques. Mais nous avons dû introduire quatre paramètres qui sont les coordonnées cartésiennes et les coordonnées cylindriques afin de représenter de façon optimale toutes les contraintes. On ne peut en effet pas exprimer une contrainte de distance par des coordonnées cartésiennes et inversement on ne peut pas représenter une zone rectangle par des coordonnées cylindriques.

Mais en revanche une contrainte de distance limite les valeurs possibles pour les coordonnées cartésiennes. De même tout type de contrainte a des conséquences sur les autres. Et le degré de liberté est de deux, donc dès que deux paramètres sont connus les autres sont fixés. Il faut donc déduire de toute combinaison de un ou deux paramètres les conséquences sur les autres paramètres. Nous avons quatre paramètres, donc dix combinaisons de 1 ou 2 paramètres, ce qui correspond au nombre de cas à examiner. Nous donnons ici les équations pour des valeurs précises, nous verrons par la suite le passage aux valeurs floues.

A) Equations pour des valeurs précises

Lorsqu'une variable est non précisée dans les équations suivantes, on sait simplement qu'elle appartaient à son domaine de définition. Nous rappelons les domaines de définition pour chaque variable :

$$\begin{cases} x \in \left]-\infty,+\infty\right[\\ y \in \left]-\infty,+\infty\right[\\ d \in \left[0,+\infty\right[\\ \theta \in \left[0,2\pi\right[\end{cases}$$

Cas 1 : si x seule est connue, on a :

$$\begin{cases} d \in \left[|x|,+\infty\right[\\ \theta \in \left[0,\pi/2\right] \cup \left[3\pi/2,2\pi\right] \text{ si } x \geq 0 \text{ et } \theta \in \left[\pi/2,3\pi/2\right] \text{ si } x \leq 0 \end{cases}$$

Cas 2 : si y seule est connue, on a :

$$\begin{cases} d \in \left[|y|,+\infty\right[\\ \theta \in \left[0,\pi\right] \text{ si } y \geq 0 \text{ et } \theta \in \left[\pi,2\pi\right] \text{ si } y \leq 0 \end{cases}$$

Cas 3 : si d seule est connue, on a :

$$\begin{cases} x \in [-d, d] \\ y \in [-d, d] \end{cases}$$

Cas 4 : si θ seule est connue, on a :

$$\begin{cases} x \in \Re^{+} \text{ si } \theta \in [0, \pi/2] \cup [3\pi/2, 2\pi] \text{ et } x \in \Re^{-} \text{ si } \theta \in [\pi/2, 3\pi/2] \\ y \in \Re^{+} \text{ si } \theta \in [0, \pi] \text{ et } y \in \Re^{-} \text{ si } \theta \in [\pi, 2\pi] \end{cases}$$

Cas 5 : si x et y sont connues, on a :

$$\begin{cases} d = \sqrt{x^2 + y^2} \\ \theta = \operatorname{Arc}\tan\dfrac{y}{x} \text{ si } (x, y) \in \Re^{+^*} \times \Re^{+} \text{ et } \theta = \pi/2 + \operatorname{Arc}\tan\dfrac{y}{x} \text{ si } (x, y) \in \Re^{-^*} \times \Re^{+} \\ \text{et } \theta = \pi + \operatorname{Arc}\tan\dfrac{y}{x} \text{ si } (x, y) \in \Re^{-^*} \times \Re^{-} \text{ et } \theta = 3\pi/2 + \operatorname{Arc}\tan\dfrac{y}{x} \text{ si } (x, y) \in \Re^{+^*} \times \Re^{-} \\ \text{et } \theta = \pi/2 \text{ si } (x, y) \in \{0\} \times \Re^{+} \text{ et } \theta = 3\pi/2 \text{ si } (x, y) \in \{0\} \times \Re^{-} \end{cases}$$

Cas 6 : si x et d sont connues, on a :

$$\begin{cases} y = 0 \text{ si } x \in \{d, -d\} \text{ et } y \in \left\{\sqrt{\left|d^2 - x^2\right|}, -\sqrt{\left|d^2 - x^2\right|}\right\} \text{ si } d^2 - x^2 \neq 0 \\ \theta = 0 \text{ si } d = 0 \text{ et } \theta \in \left\{\operatorname{Arc}\cos\dfrac{x}{d}, 2\pi - \operatorname{Arc}\cos\dfrac{x}{d}\right\} \text{ si } d \neq 0 \end{cases}$$

Cas 7 : si x et θ sont connues, on a :

$$\begin{cases} y \in [0, +\infty[\text{ si } (x, \theta) \in \{(0, \pi/2)\} \text{ et } y \in]-\infty, 0] \text{ si } (x, \theta) \in \{(0, 3\pi/2)\} \text{ et} \\ y = x \tan\theta \text{ si } \theta \neq \{\pi/2, 3\pi/2\} \\ d = \dfrac{x}{\cos\theta} \text{ si } \cos\theta \neq 0 \end{cases}$$

Cas 8 : si y et d sont connues, on a :

$$\begin{cases} x = 0 \text{ si } y \in \{d,\text{-}d\} \text{ et } x \in \left\{\sqrt{\left|d^2 - y^2\right|}, -\sqrt{\left|d^2 - y^2\right|}\right\} \text{ si } d^2 - y^2 \neq 0 \\ \theta = 0 \text{ si } d = 0 \text{ et } \theta \in \left\{\operatorname{Arc}\sin\dfrac{y}{d}, \left(\pi/2 + \operatorname{Arc}\sin\dfrac{y}{d}\right)\right\} \text{ si } y/d > 0 \\ \text{et } \theta \in \left\{\left(2\pi + \operatorname{Arc}\sin\dfrac{y}{d}\right), \left(\pi - \operatorname{Arc}\sin\dfrac{y}{d}\right)\right\} \text{ si } y/d < 0 \end{cases}$$

Cas 9 : si y et θ sont connues, on a :

$$\begin{cases} x = 0 \text{ si } \theta \in \{\pi/2, 3\pi/2\} \text{ et } x \in [0,+\infty[\text{ si } \theta = 0 \text{ et } x \in]-\infty,0] \text{ si } \theta = \pi \\ \text{et } x = \dfrac{y}{\tan\theta} \text{ si } \tan\theta \neq 0 \\ d = \dfrac{y}{\sin\theta} \text{ si } \sin\theta \neq 0 \end{cases}$$

Cas 10 : si d et θ sont connues, on a :

$$\begin{cases} x = d\cos\theta \\ y = d\sin\theta \end{cases}$$

B) Extension au cas flou

Ces équations sont issues de la géométrie classique, mais pour les appliquer à notre modèle nous les avons étendues au cas où les variables x, y, d et θ sont des nombres flous X, Y, D et Θ. Nous nous plaçons dans le cas général d'une variable a (représentée par le nombre flou A) exprimée en fonction d'une autre variable b (représentée par le nombre flou B). Les conclusions des équations sont de deux formes :

- a ∈ e, où e est soit un intervalle constant soit un intervalle construit avec b
- a = f(b) où f est une fonction

Avec l'extension au flou, e devient un ensemble flou E s'il dépend de b. Si e est de la forme [b,+∞[, alors E est l'ensemble "possiblement supérieur à B" dont on a rappelé la définition en II.1.4B)a) p.25. Si e est un intervalle constant, E est un ensemble flou réduit à un intervalle classique. Pour calculer f(b) on applique le principe d'extension et on obtient un nombre flou F. Pour le détail de l'application du principe d'extension aux fonctions rencontrées dans les équations, voir V.5.2C) p.226. Les deux équations précédentes prennent donc la forme :

- $A \subseteq E$
- $A = F \Leftrightarrow [(A \subseteq F) \wedge (F \subseteq A)]$

Lors de la propagation, ces équations ne sont utilisées que pour réduire l'ensemble flou A, donc pour l'égalité A = F seule la première partie de l'équivalence ($A \subseteq F$) est utile. Les deux équations ont finalement la forme d'une inclusion. $A \subseteq E$ se traduit par $A'=A\cap E$ où A' est le nouvel ensemble flou de la variable a, après propagation selon cette équation.

Nous avons aussi un certain nombre d'expressions conditionnelles dans ces équations. Dans le cas flou, les conditions ne sont pas booléennes mais peuvent être vérifiées avec un certain degré. Si la condition est partiellement vérifiée, la conclusion est aussi partiellement vraie, donc on lui applique un degré maximum égal au degré de vérification de la condition.

On trouve plusieurs types de conditions. Une première condition rencontrée dans le cas 1 est : x >0. Si X représente le nombre flou x, la condition est vérifiée avec un degré qui est la hauteur de l'ensemble $X\cap]0,+\infty[$. Une condition exclusive comme dans le cas 6 "$d\neq0$" est vérifiée avec un degré égal à la hauteur de $(D\cap]0,+\infty[)\cup(D\cap]-\infty,0[)$. Pour une condition disjonctive du type "x vérifie A ou B" (cas 4) on prend le degré maximum de "x vérifie A" et "x vérifie B".

Certains paramètres sont liés par une formule complexe à d'autres paramètres comme dans le cas 7, avec des conjonctions de formules du type : "si A alors x vaut V_A et si B alors x vaut V_B". Le résultat est l'union de chacune des conclusions induites séparément par "si A alors x vaut V_A" et par "si B alors x vaut V_B".

C) Calcul des opérateurs flous

Toutes ces équations utilisent aussi des opérateurs sur les nombres flous qui n'ont pas encore été définis jusqu'ici. Ces opérateurs sont toutes les fonctions trigonométriques ainsi que les fonctions inverse, produit, racine et carré. Il faut aussi définir la valeur absolue et le modulo 2π pour des nombres flous, puisque ces opérations sont utilisées pour réaliser la composition, respectivement dans les graphes de distances et de directions.

Ces fonctions ne sont pas forcément monotones sur tout leur domaine de définition. On ne peut donc plus appliquer la méthode de calcul de fonctions présentée en III.3.2B)c)ii) p.97, puisque ce calcul est valable uniquement pour des fonctions monotones. Pour le calcul de ces fonctions nous allons adopter une modélisation par α-coupes. On rappelle dans ce cas le calcul de f(A), avec $[W]_\alpha$ la α-coupe de W de niveau α [Kaufmann 91] :

Si f est continue et croissante, alors pour $A_\alpha=[a_\alpha,b_\alpha]$ on a $[f(A)]_\alpha=[f(a_\alpha),f(b_\alpha)]$.

Si f est continue et décroissante, alors $[f(A)]_\alpha=[f(b_\alpha),f(a_\alpha)]$.

Si f est continue et non monotone, alors $[f(A)]_\alpha=[\inf_{x\in[a_\alpha,b_\alpha]} f(x), \sup_{x\in[a_\alpha,b_\alpha]} f(x)]$.

En dehors des fonctions inverse, tangente et modulo 2π, toutes les fonctions que nous devons définir sont continues et monotones sur des sous-domaines du domaine de définition. Par exemple la fonction cosinus est croissante sur $[\pi,2\pi]$ et décroissante sur $[0,\pi]$. Plus généralement, une fonction f continue et monotone sur des sous domaines peut se calculer en appliquant le théorème suivant :

Théorème : Soit f une fonction continue définie sur le domaine D, croissante sur D_1 et décroissante sur D_2 avec $D=D_1\cup D_2$. Soit le nombre flou A et le réel $\alpha\in[0,1]$ tels que $A_\alpha\cap D_1=[a_1,b_1]$ et $A_\alpha\cap D_2=[a_2,b_2]$. Alors $[f(A)]_\alpha=[\min(f(a_1),f(b_2)),\max(f(b_1),f(a_2))]$.

Démonstration : f est continue donc on a $[f(A)]_\alpha=[\inf_{x\in[a_\alpha,b_\alpha]} f(x), \sup_{x\in[a_\alpha,b_\alpha]} f(x)]$ avec $A_\alpha=[a_\alpha,b_\alpha]$.

Or $D=D_1\cup D_2$ et $A_\alpha\subseteq D$ donc $A_\alpha=A_\alpha\cap D=A_\alpha\cap(D_1\cup D_2)=(A_\alpha\cap D_1)\cup(A_\alpha\cap D_2)=[a_1,b_1]\cup[a_2,b_2]$. Donc $[f(A)]_\alpha=[\min(\inf_{x\in[a_1,b_1]} f(x), \inf_{x\in[a_2,b_2]} f(x)),\max(\sup_{x\in[a_1,b_1]} f(x), \sup_{x\in[a_2,b_2]} f(x))$. f est croissante sur D_1 donc $\inf_{x\in[a_1,b_1]} f(x)=f(a_1)$ et $\sup_{x\in[a_1,b_1]} f(x)=f(b_1)$. F est décroissante sur D_2 donc $\inf_{x\in[a_2,b_2]} f(x)=f(b_2)$ et $\sup_{x\in[a_2,b_2]} f(x)=f(a_2)$. Donc $[f(A)]_\alpha=[\min(f(a_1),f(b_2)),\max(f(b_1),f(a_2))]$. ❒

Reprenons l'exemple de la fonction cosinus. Soit par exemple la α-coupe $[a,b]$ avec a dans $[0,\pi]$ et b dans $[\pi,2\pi]$. On a alors $\cos([a,b])=[\cos\pi,\max(\cos a,\cos b)]$. On procède de même pour toutes les fonctions exceptées les fonctions inverse et tangente. On remarque que l'application de ces fonctions sur des ensembles non disjonctifs donne un résultat non disjonctif.

Pour les fonctions inverse, tangente et modulo 2π il existe des points de discontinuité qui sont aussi les points de discontinuité du domaine de définition. La fonction inverse est discontinue en 0, tangente est discontinue en $\pi/2$ et $3\pi/2$ et modulo 2π est discontinue en tous les multiples de 2π. Le résultat de la fonction est alors obtenu par l'union des résultats obtenus pour chacun des sous-domaines de définition. Ceci en utilisant le théorème suivant :

Théorème : Soit f une fonction définie sur $D=D_1\cup D_2$, A un ensemble flou de fonction d'appartenance μ_A tel que $A=A\cap D$. f(A) est l'ensemble flou de fonction d'appartenance $\mu_{f(A)}$ obtenu par application de f sur A. Alors $f(A)=f(A\cap D_1)\cup f(A\cap D_2)$.

Démonstration : On applique le principe d'extension à f, ce qui donne :

$$\mu_{f(A)}(y)=\sup_{\substack{y=f(x)\\ x\in A}} \mu_A(x)$$

$$=\sup_{\substack{y=f(x)\\ x\in A\cap(D_1\cup D_2)}} \mu_A(x)=\max\Big(\sup_{\substack{y=f(x)\\ x\in A\cap D_1}} \mu_A(x), \sup_{\substack{y=f(x)\\ x\in A\cap D_2}} \mu_A(x)\Big)$$

$\mu_{D_i}(x) = 1$ si $x \in D_i$ et 0 sinon donc $\mu_{A\cap D_i}(x) = \min(\mu_A(x), \mu_{D_i}(x)) = \mu_A(x)$ si $x \in D_i$

Donc $\mu_{f(A)}(y) = \max(\sup\limits_{\substack{y=f(x)\\ x\in A\cap D_1}} \mu_{A\cap D_1}(x), \sup\limits_{\substack{y=f(x)\\ x\in A\cap D_2}} \mu_{A\cap D_2}(x))$

Toujours d'après le principe d'extension, $\sup\limits_{\substack{y=f(x)\\ x\in A\cap D_i}} \mu_{A\cap D_i}(x) = f(A\cap D_i)$.

Donc $\mu_{f(A)}(y) = \max(f(A\cap D_1), f(A\cap D_2)) \Rightarrow f(A)=f(A\cap D_1)\cup f(A\cap D_2)$ ❒

De plus les fonctions tangente, inverse et modulo 2π sont monotones sur chacun de leurs sous-domaines de définition, donc leur calcul sur chacun des sous-domaines est aisé en passant par les α-coupes. X étant un ensemble flou, le calcul de l'inverse, de la tangente et du modulo 2π peuvent donc se décomposer ainsi :

$$\frac{1}{X} = \frac{1}{X \cap \left]-\infty,0\right[} \cup \frac{1}{X \cap \left]0,+\infty\right[}$$

$$\tan(X)=\tan(X\cap[0,\pi/2])\cup\tan(X\cap[\pi/2,3\pi/2])\cup\tan(X\cap[3\pi/2,2\pi])$$

$$A[2\pi]=(A\cap[0,2\pi])\cup\{(A\cap[2\pi,4\pi])-2\pi\}\cup\{(A\cap[4\pi,6\pi])-4\pi\}\ldots$$

Avec ces fonctions le résultat peut être disjonctif même si les ensembles en entrée sont non disjonctifs. On peut prendre l'exemple de $A=[3\pi/2,5\pi/2]$: $A[2\pi]=[3\pi/2,2\pi]\cup[0,\pi/2]$. De même les équations des cas 6 et 8 peuvent introduire des disjonctions. On a vu que le calcul doit passer par les α-coupes, donc les disjonctions devront être modélisées de façon explicite. On aura donc des ensembles de la forme "A ou B" avec A et B des ensembles flou convexes. On pourra simplifier ces ensembles en appliquant :

Si $[\forall\ \alpha$ tel que $A_\alpha\neq\emptyset$ et $B_\alpha\neq\emptyset$ on a $A_\alpha\cap B_\alpha\neq\emptyset]$ alors $A\cup B$ est convexe.

V.5.3 Algorithme global de propagation

A) Description

L'algorithme global de propagation consiste donc à appliquer la consistance de chemins sur chacun des graphes séparément, à vérifier les contraintes de zone et à appliquer les contraintes induites d'un ou deux formalismes dans l'autre. On répète ces opérations tant que des contraintes sont modifiées. On effectue uniquement les opérations qui font intervenir des contraintes récemment modifiées. L'algorithme se termine puisque l'on ne peut que réduire les contraintes qui sont modélisées sur un univers discret et avec un nombre de α-coupes fini.

On appelle G_p, G_d et G_{dir} respectivement les graphes de positions, de distances et de directions. Ils comportent tous les mêmes nœuds et sont complets. Les contraintes de G_p, G_d, G_{dir} entre les nœuds i et j sont respectivement (X_{ij},Y_{ij}), D_{ij}, Dir_{ij}. Voici donc l'algorithme global.

Algorithme global de propagation des contraintes spatiales :

- 0. Tant que des contraintes sont modifiées et non vides faire :
- 1. Appliquer l'algorithme de consistance de chemins sur G_p, G_d et G_{dir}.
- 2. Appliquer les contraintes de zone dans chacun des graphes G_p, G_d et G_{dir}.
- 3. Pour chaque quadruplet de contraintes (X_{ij}, Y_{ij}, D_{ij}, Dir_{ij}), faire :
- 4. Déduire Dir_{ij} de X_{ij}, Y_{ij} et D_{ij} (cas 1 et 2)
- 5. Déduire D_{ij} de X_{ij}, Y_{ij} et Dir_{ij} (cas 1 et 2)
- 6. Déduire Y_{ij} de X_{ij}, D_{ij} et Dir_{ij} (cas 3 et 4)
- 7. Déduire X_{ij} de Y_{ij}, D_{ij} et Dir_{ij} (cas 3 et 4)
- 8 Déduire Dir_{ij} des combinaisons (X_{ij},Y_{ij}), (X_{ij},D_{ij}), et (Y_{ij},D_{ij}) (cas 5, 6 et 8)
- 9. Déduire D_{ij} des combinaisons (X_{ij},Y_{ij}), (X_{ij},Dir_{ij}), et (Y_{ij},Dir_{ij}) (cas 5, 7 et 9)
- 10. Déduire Y_{ij} des combinaisons (X_{ij},D_{ij}), (X_{ij},Dir_{ij}), et (D_{ij},Dir_{ij}) (cas 6, 7 et 10)
- 11. Déduire X_{ij} des combinaisons (Y_{ij},D_{ij}), (Y_{ij},Dir_{ij}), et (D_{ij},Dir_{ij}) (cas 8, 9 et 10)
- 12. Si une contrainte est vide le problème est inconsistant et arrêt.
- 13. Sinon retourner les 3 graphes G_p, G_d et G_{dir}.

B) Non complétude

Nous allons voir que contrairement à ce que l'on avait pour le graphe temporel, l'algorithme de propagation pour les contraintes spatiales n'est pas complet. On obtient bien le graphe minimal pour le graphe de positions mais pas pour les graphes de distances et de directions. Ceci est du à la non-distributivité de la composition sur l'intersection dans ces cas là.

Théorème : Pour des distances ou des directions, la composition est non-distributive sur l'intersection.

Démonstration : Voici un contre-exemple prouvant la non-distributivité dans le cas de distances. Soient les distances d_1=5 ; d_2=3 et d_3=10 :

$d_2 \cap d_3 = \varnothing$ donc $d_1 o (d_2 \cap d_3) = \varnothing$

$d_1 o d_2 = [2,8]$ et $d_1 o d_3 = [5,15]$ donc $(d_1 o d_2) \cap (d_1 o d_3) = [5,8]$

Donc on a bien un exemple de non-distributivité $(d_1 o d_2) \cap (d_1 o d_3) \neq d_1 o (d_2 \cap d_3)$.

De même on peut trouver un contre-exemple avec les direction $\theta_1=\pi/4$, $\theta_2=\pi/3$ et $\theta_3=\pi/2$.

$\theta_2 \cap \theta_3 = \varnothing$ donc $\theta_1 o (\theta_2 \cap \theta_3) = \varnothing$

$\theta_1 o \theta_2=[\pi/4,\pi/3]$ et $\theta_1 o \theta_3=[\pi/4,\pi/2]$ donc $(\theta_1 o \theta_2)\cap(\theta_1 o \theta_3)=[\pi/4,\pi/3]$

Ceci est bien un exemple de non distributivité $(\theta_1 o \theta_2)\cap(\theta_1 o \theta_3)\neq\theta_1 o(\theta_2\cap\theta_3)$. ❐

Théorème : L'algorithme de consistance de chemins sur le graphe de distances n'est pas complet.

Démonstration : Nous considérons le graphe suivant de 4 nœuds A, B, C et D avec la contrainte de distance 1 entre tout couple de nœuds.

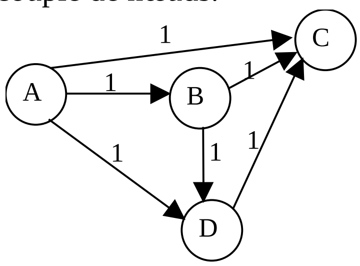


Fig. 133 : exemple de graphe chemins-consistant et non consistant

Ce graphe n'est pas modifié par l'algorithme de propagation, il est donc bien chemins-consistant. Mais il n'est pas consistant. On peut en effet démontrer par une interprétation géométrique qu'il n'existe pas de solution en dimension 2 vérifiant ces contraintes (en dimension 3, la solution est un tétrahèdre régulier). A, B et C sont en effet trois points équidistants donc ils forment un triangle équilatéral. D est aussi équidistant de ces trois points donc il est à l'intersection des médiatrices du triangle ABC, c'est le centre de gravité G du triangle. Mais la distance AG vaut alors 2/3 et les contraintes AD=BD=CD=1 ne peuvent pas être vérifiées. Il n'existe pas de solution, on a bien un graphe chemins-consistant et non consistant. ❐

On pourrait sans doute trouver le même type d'exemple de non consistance pour le graphe de directions.

Etant donné que l'on ne peut pas assurer la complétude de l'algorithme dans le cas de contraintes de distances ou de directions non disjonctives, il est inutile d'ajouter des traitements spécifiques pour les contraintes disjonctives. Même si l'on séparait un graphe disjonctif en graphes simples afin d'appliquer la propagation sur ces graphes simples, on n'est pas assuré d'avoir le graphe minimal correspondant à chaque graphe simple et l'union des graphes résultants n'est donc pas forcément le graphe minimal. On applique donc l'algorithme de propagation tel quel sur chaque graphe en considérant les contraintes disjonctives comme n'importe quelle contrainte.

C) Consistance de chemins

Par construction, nous avons tout de même trois graphes qui sont séparément 3-consistants. De plus, les graphes sont complets ou peuvent être complétés donc la 3-consistance nous assure la consistance de chemins dans chacun des graphes. On peut même affirmer que le graphe de positions est minimal s'il ne comporte pas de contrainte disjonctive. La composition de contraintes cartésiennes simples est en effet distributive sur l'intersection, donc on peut montrer la minimalité dans le cas de contraintes non disjonctives de même qu'on l'avait fait pour les contraintes temporelles. Mais si les contraintes résultantes sont disjonctives on se contente de la consistance de chemins puisque les graphes de distances et de directions ne sont pas forcément minimaux.

De plus la traduction des contraintes d'un formalisme à l'autre est optimale. Une traduction est optimale si et seulement si cette traduction n'élimine aucune des solutions présentes avant la traduction (1) et si elle élimine bien toutes les valeurs impossibles pour un paramètre (2). Soit un vecteur de coordonnées cartésiennes (x,y) et de coordonnées cylindrique (d,θ). Ces quatre paramètres doivent alors vérifier les équations suivantes :

$$\begin{cases} d^2 = x^2 + y^2 \\ \tan\theta = y / x \\ x = d\cos\theta \\ y = d\sin\theta \end{cases}$$

Ces équations décrivent le passage des coordonnées cartésiennes aux cylindriques et inversement. On peut montrer que ces seules équations induisent toutes les contraintes des cas 1 à 10 présentées en V.5.2 p.223. On le démontre ici pour les équations du cas 1 et la démonstration se fait sur le même principe pour tous les autres cas.

$$\begin{cases} d^2 = x^2 + y^2 \Rightarrow d^2 > x^2 \text{ et } d > 0 \text{ d'après sa définition donc } d > |x| \Rightarrow d \subseteq \left[|x|, +\infty\right[\\ \text{si } x \geq 0 \text{ alors } \cos\theta = x/d \Rightarrow \cos\theta \geq 0 \Rightarrow \theta \subseteq \left[0, \pi/2\right] \cup \left[3\pi/2, 2\pi\right] \\ \text{si } x \leq 0 \text{ alors } \cos\theta = x/d \Rightarrow \cos\theta \leq 0 \Rightarrow \theta \subseteq \left[\pi/2, 3\pi/2\right] \end{cases}$$

On a donc bien toutes les contraintes induites du cas 1. Les contraintes induites n'éliminent pas de solution, elles rendent les contraintes plus explicites. On a donc bien montré la partie (1) de la définition d'une "traduction optimale".

La partie (2) de la définition consiste à dire que toute valeur induite par une contrainte correspond à au moins un point qui satisfait effectivement cette contrainte. Prenons l'exemple d'une contrainte d'abscisse relative $x \in X$. Cette contrainte induit les contraintes du cas 1 suivantes :

$$\begin{cases} d \subseteq \left[|X|,+\infty\right[\\ \theta \subseteq \left[0,\pi/2\right] \cup \left[3\pi/2,2\pi\right] \text{ si } x \geq 0 \text{ et } \theta \subseteq \left[\pi/2,3\pi/2\right] \text{ si } x \leq 0 \end{cases}$$

Il faut alors montrer que pour toute valeur de distance $d \subseteq D = \left[|X|,+\infty\right[$ il existe un vecteur de norme d et d'abscisse x qui vérifie aussi la contrainte $x \in X$ avec au moins le même degré. Soit d_1 une distance telle que $\mu_D(d_1)=\alpha$, nous cherchons une valeur d'abscisse x_0 telle que $\mu_X(x_0)\geq\alpha$. Or nous avons :

$$D=[|X|,+\infty[$$

$$\mu_{|X|}(x) = \sup_{x=|y|} \mu_X(y) = \max(\mu_X(x),\mu_X(-x))$$

$$\mu_D(x) = \mu_{[|X|+\infty[}(x) = \sup_{y\leq x} \mu_{|X|}(y)$$

$$\mu_D(d_1) = \alpha = \sup_{y\leq d_1} \mu_{|X|}(y) \Rightarrow \exists\, x_1 \text{ tel que } \mu_{|X|}(x_1) = \alpha$$

$$\Rightarrow \max(\mu_X(x_1),\mu_X(-x_1)) = \alpha$$

$$\Rightarrow \exists\, x_0 \text{ tel que } \mu_X(x_0) = \alpha \text{ avec } x_0 = x_1 \text{ ou } x_0 = -x_1$$

On peut faire le même type de démonstration pour toutes les contraintes induites. Notre traduction est donc optimale, il n'y a pas de perte d'information : aucune solution n'est éliminée après la traduction et la contrainte induite ne peut pas être plus restrictive sans perdre de solution.

V.6 Conclusion

L'intérêt de notre modèle est de permettre la prise en compte de plusieurs aspects simultanément. Il traite à la fois des contraintes de distances, des contraintes de positions et des contraintes de direction. Dans les modèles présents dans la littérature, la plupart du temps on ne traite qu'un seul aspect, alors qu'il est clair que tous les aspects interagissent. Ils sont dédiés à une forme particulière d'information.

Le modèle de Dutta V.1.1C)d)i) p.193 est le seul à prendre en compte à la fois des informations de position, de distance et de direction. Mais il traduit directement toutes ces informations en une information de position en coordonnées cartésiennes. Il y a donc une perte d'information à ce niveau. Il y a d'autant plus de perte d'information que l'information au départ est incomplète, ce qui est une caractéristique de notre application à la reconnaissance d'activités militaires.

Nous effectuons aussi des traductions d'un formalisme à un autre, mais l'information n'est pas perdue car elle reste présente sous ses trois formes. Nous allons illustrer cette différence avec un exemple très simple. Soit une information : "B est à 10 mètres de A". Si on reprend le formalisme de Dutta on a une seule contrainte entre A et B, contrainte vectorielle V(A,B)=([0,10],[0,10]). Dans notre modèle, on aura une contrainte de position P(A,B)=V(A,B), une contrainte de distance D(A,B)=10 et une contrainte de direction quelconque. On considère que A est l'origine du repère de coordonnées (0,0). B a pour coordonnées (2,2) est accepté dans le modèle de Dutta alors que dans notre modèle il est rejeté car ne vérifiant évidemment pas la contrainte de distance. Il n'y a pas eu de perte d'information.

Même si on ne peut pas être sûr d'avoir des graphes minimaux après la propagation, ces graphes contiennent plus d'information que le seul graphe de position de Dutta qui est pourtant minimal. De plus, on a tout de même la consistance de chemins dans chacun des graphes, ce qui constitue une bonne approximation des graphes minimaux. La consistance de chemins permet de détecter un bon nombre d'inconsistances. Et chaque graphe ou couple de graphes est traduit de façon optimale dans les autres graphes, donc chaque graphe bénéficie du maximum d'informations exprimables dans le formalisme correspondant.

Par rapport à une simple vérification des contraintes comme cela était fait jusqu'ici, on exclut certains scenarii de la reconnaissance beaucoup plus tôt. Même si l'on n'est pas sûr de garder uniquement les gabarits encore réalisables, on en élimine un certain nombre, donc la reconnaissance est beaucoup plus efficace.

La propagation n’est certes pas aussi efficace que pour les gabarits temporels. Ceci n’est pas étonnant du fait de la complexité bien supérieure de la modélisation spatiale par rapport à la modélisation temporelle.

VI Conclusion et perspectives

La reconnaissance de scenarii en général est une comparaison entre une réalité observée et des scenarii prédéfinis. Ces scenarii prédéfinis reflettent la connaissance de l'expert sur les phénomènes concernés. Ils sont souvent difficiles à décrire de façon précise : un scénario doit regrouper plusieurs déroulements possibles d'un même phénomène global (une activité militaire en l'occurrence). Il doit donc être assez général pour englober tous ces déroulements semblables tout en étant assez précis pour exclure des phénomènes proches dans leur déroulement mais différents dans leur interprétation à un plus haut niveau. La frontière entre les phénomènes à englober et ceux à exclure est donc assez difficile à déterminer.

Un scénario comporte un certain nombre de faits obligatoires. Mais la liste des faits ne suffit pas à elle-seule à reconnaître le scénario. Les paramètres liés à ces faits doivent aussi vérifier un certain nombre de contraintes. Même si les faits sont obligatoires, les contraintes à vérifier sont vagues et peuvent être plus ou moins bien vérifiées, ce qui permet de distinguer plusieurs niveaux de reconnaissance. Deux phénomènes observés peuvent être tout à fait identiques du point de vue de la liste des faits les composant mais totalement distincts dans leur interprétation du fait de leur organisation dans le temps ou dans l'espace.

Les contraintes apparaissent donc primordiales pour la représentation d'un scénario. Une fois que les faits bruts sont reconnus, ce sont ces contraintes qui permettent d'aiguiller et d'évaluer la reconnaissance. Il est donc important de bien les représenter. Ce sont ces contraintes qui établissent la frontière entre les phénomènes correspondant ou non au scénario. On a vu que cette frontière était vague, il est donc important d'introduire des contraintes floues pour la représenter. L'originalité de nos travaux réside en particulier dans l'introduction de cette souplesse de modélisation des scenarii.

Cette souplesse dans la modélisation se traduit naturellement par plus de finesse dans l'évaluation de la reconnaissance. Puisque les contraintes sont floues, les phénomènes observés satisfont plus ou moins les contraintes et donc on est plus ou moins proche d'un scénario donné. Le résultat de la reconnaissance est donc plus proche de ce que pourrait conclure un expert observant la scène : "il y a beaucoup de chances pour que l'on soit en présence de tel scénario".

De plus, dans notre cas particulier de reconnaissance d'activités militaires, nous avions aussi besoin de représenter l'imprécision dans les observations elles-mêmes. La réalité peut donc se représenter exactement de la même façon qu'un scénario. La modélisation sous forme de graphes de contraintes semblant être la plus efficace pour la propagation et la plus expressive, nous avons donc modélisé les scénario et la réalité par des graphes de contraintes floues.

Nous avons donc été amenés à étendre le formalisme TCSP de représentation des contraintes temporelles à des contraintes floues. Nous avons étudié plus précisément le pouvoir d'expression d'une telle représentation ainsi que le calcul du graphe minimal. Le graphe minimal est en effet fondamental pour la reconnaissance puisqu'il aboutit à une propagation optimale ; on ne peut pas extraire plus d'information de nos connaissances et de nos observations sans hypothèse supplémentaire.

Les FCSP étaient déjà une extension floue des CSP, mais nous avons élargi l'emploi des FCSP à une nouvelle catégorie de problèmes. Jusqu'ici en effet les FCSP n'avaient été envisagés que sous un seul point de vue : la recherche d'une solution particulière la meilleure possible à un problème décrit par des contraintes floues. Dans notre cas de reconnaissance de scenarii il ne s'agit pas de trouver une solution mais de prédire si les observations faites permettent de respecter un modèle représenté par ses contraintes floues.

Nous avons appliqué notre technique à des contraintes temporelles dans un premier temps puis nous l'avons étendu à des contraintes spatiales. Le choix de ce type de contraintes est lié à la nature des scenarii. Ces contraintes sont en effet prépondérantes dans les activités militaires. Il s'agit dans les deux cas de contraintes binaires. Pour des contraintes unaires le traitement est beaucoup plus simple car il suffit simplement de vérifier les contraintes et il n'y a pas de propagation. C'est en effet l'étape de propagation qui est délicate.

Elle n'est pas aussi performante dans le cas des contraintes spatiales que dans le cas du temporel. Les contraintes temporelles sont en effet beaucoup plus simples : elles sont unidimensionnelles et peuvent toutes s'exprimer sous la forme de durées. Pour les relations spatiales en revanche, il n'existe pas une forme de contrainte qui permette d'exprimer toutes les relations possibles. Nous avons donc été obligés d'employer plusieurs formalismes simultanément. Notre système permet ainsi de prendre en compte beaucoup plus de contraintes que les autres : il tient compte des interactions entre distances, positions et directions. C'est le seul à le proposer à notre connaissance.

Même si notre propagation n'est pas optimale, elle permet de déuire plus d'information que des modèles plus réducteurs qui ne prennent en compte qu'un seul type de contraintes spatiales. De plus, on a tout de même la consistance de chemins dans chacun des graphes, ce qui constitue une bonne approximation des graphes minimaux. La consistance de chemins permet de détecter un bon nombre d'inconsistances.

Par rapport à une simple vérification des contraintes comme cela était fait jusqu'ici, on exclut certains scenarii de la reconnaissance beaucoup plus tôt. Même si l'on n'est pas sûr de garder uniquement les gabarits encore réalisables, on en élimine un certain nombre, donc la reconnaissance est beaucoup plus efficace. Il serait intéressant cependant de réaliser des expérimentations avec une bibliothèque de gabarits établie par des opérationnels et avec un scénario réaliste. Cela permettrait de tester pratiquement la complexité des algorithmes.

Mais que se passerait-il avec des contraintes n-aires ? Il serait en effet intéressant de généraliser notre méthode à des contraintes impliquant plus de deux éléments du scénario. Cela permettrait typiquement de décrire la forme géométrique des objets. Pour exprimer que 3 points forment un triangle équilatéral on a en effet besoin d'une contrainte ternaire. Il serait aussi intéressant d'avoir une caractérisation du type de contraintes pour lequel ce genre de modélisation est possible. Ce travail de généralisation et de plus grande formalisation nous semble prometteur pour une application à un plus grand nobre de problèmes.

Une autre approche pour poursuivre ce travail serait une aide à l'élaboration des scenarii. Nous avons abordé ce problème pour le choix des mesures d'agrégation qui dépend typiquement de l'interprétation faite par l'expert. La partie recueil d'expertise est souvent un travail très difficile et le résultat constitue une première vue synthétique mais demande à être raffiné. On pourrait donc imaginer que les scenarii initiaux sont obtenus par cet expertise mais que le système raffine ces scenarii par un apprentissage supervisé. Il est en effet plus facile de faire dire à un expert quel scénario il a vu se dérouler plutôt que de lui faire décrire ce scénario dans l'absolu. Les paramètres à régler sont les contraintes et les formes d'agrégation à employer. Ces points relèvent en effet d'une interprétation inconsciente difficile voire impossible à exprimer.

Finalement, notre système réalise de la fusion de données puisqu'il extrait des informations nouvelles à partir d'informations de différents types et de différentes sources : il fusionne des observations avec des modèles de scenarii. Cette fusion s'établit à deux niveaux : d'une part au niveau des contraintes, qu'elles soient temporelles ou spatiales, et d'autre part au niveau de la structure du gabarit. Dans la première phase la propagation des contraintes réalise leur fusion. Dans la seconde phase, les opérateurs d'agrégation sont un moyen de fusionner des informations non structurées avec un modèle donné sous la forme d'un arbre.

Notre système peut s'appliquer à d'autres domaines d'application fonctionnant toujours sur un mécanisme de reconnaissance de scenarii avec des modèles de scenarii structurés et contraints. Il pourrait parfaitement s'adapter à de la supervision, à du diagnostic de pannes ou encore au diagnostic médical. Pour ce dernier exemple la description des symptômes et de leurs caractéristiques temporelles est entâchée de beaucoup d'imprécisions, l'apport des contraintes floues pourrait donc être notable.

VII Annexes

VII.1 Reconnaissance par comparaison de graphes

VII.1.1 Notions théoriques sur les graphes

Nous rappelons ici les notions relatives à la comparaison de deux graphes. Soient deux graphes G_1 et G_2 cohérents et complets. $G_1=(\mathcal{U}_1,\Re_1)$ avec $\mathcal{U}_1=(U_{11},U_{12},...,U_{1p})$ est l'ensemble des variables et $\Re_1=\{r_{ij}^1/0\leq i,j\leq p\}$ est l'ensemble des contraintes. $G_2=(\mathcal{U}_2,\Re_2)$ avec $\mathcal{U}_2=(U_{21},U_{22},...,U_{2q})$ et $\Re_2=\{r_{ij}^2/0\leq i,j\leq q\}$. $\mathcal{W}=\mathcal{U}_1\cap\mathcal{U}_2$.

Si $\mathcal{W}=\emptyset$ alors les graphes G_1 et G_2 sont ***disjoints***. Ils n'ont aucune contrainte ayant les mêmes extrémités.

S'ils sont non disjoints, ils ont un recouvrement $I=G_1\cap G_2=G_{1\mathcal{W}}\cap G_{2\mathcal{W}}$ où $G_{1\mathcal{W}}$ (resp. $G_{2\mathcal{W}}$) sont les sous-graphes de G_1 (resp. G_2) dont les nœuds sont les éléments de $\mathcal{W}$ et les arcs ceux de G_1 (resp. G_2) ayant leurs deux extrémités dans $\mathcal{W}$. L'intersection des graphes s'obtient par intersection des contraintes de chaque arc. On distingue 4 possibilités :

Inclusion dans un sous-graphe : G_1 est inclus dans un sous-graphe de G_2 (inclusion sur les nœuds et sur les contraintes de mêmes extrémités).

Inclusion par contraintes isolées : $G_i=(\mathcal{U}_i,\Re_i)$ est un ***sous-graphe isolé*** de $G=(\mathcal{U},\Re)$ si $\mathcal{U}_i\subset\mathcal{U}$ et pour tout élément u de $\mathcal{U}_i$ il existe au moins une contrainte de $\Re_i$ qui n'est pas universelle et si toutes les contraintes de $\Re$ ayant une extrémité dans $\mathcal{U}_i$ et l'autre dans $\mathcal{U}$-$\mathcal{U}_i$ sont universelles. Tout graphe G peut être décomposé en une union de sous-graphes isolés. G_1 est inclus par composantes isolées dans G_2 ($G_1\subset^{pci}G_2$) si tout graphe de sa décomposition en sous-graphes isolés est inclus dans un sous-graphe de G_2.

Intersection simple : I est strictement inclus dans $G_{1\mathcal{W}}$ et $G_{2\mathcal{W}}$.

Intersection simple avec inclusion : $I=G_{1\mathcal{W}}$.

On peut aussi comparer les solutions de deux graphes. Les solutions $S_i=(u_{i1},u_{i2},...,u_{in})$ i=1,2 sont ***équivalentes*** si $u_{2j}-u_{1j}=\lambda\ \forall i,\ \lambda\in\Re$. Elles sont ***compatibles*** si les assignations sur leurs variables communes sont identiques.

Fusion de graphes : la fusion de deux graphes G_1 et G_2 notée $G_1 \oplus G_2$ est l'intersection de leurs sur-graphes $G_1^{\mathcal{U}}$ et $G_2^{\mathcal{U}}$ avec $\mathcal{U} = \mathcal{U}_1 \cup \mathcal{U}_2$.

VII.1.2Reconnaissance globale

Nous allons passer maintenant à l'analyse de la reconnaissance globale d'une session par rapport à un scénario. Une session est un compte-rendu du déroulement des événements avec les informations temporelles associées. On peut décrire une session Σ sous la forme d'un graphe $\Gamma(\Sigma)$ de contraintes ρ_{ij}. La reconnaissance consiste à comparer les deux graphes $\Gamma(\Sigma)$ et G(S). On parle de reconnaissance globale par opposition à reconnaissance incrémentale lorsque la reconnaissance a lieu une fois que la session est terminée ; on intègre directement tous les éléments de la session et non pas un par un au fur et à mesure de leur arrivée. On peut avoir trois types de conclusions après la reconnaissance : la session satisfait le scénario, ce qui est une association forte, la session est compatible avec le scénario, ce qui est plus faible, ou enfin la session rejette le scénario si la reconnaissance échoue.

On considère pour le scénario son graphe de contraintes G cohérent, minimal et entièrement contraint (suppression des nœuds inutiles n'ayant d'autre contrainte que la contrainte universelle). Le graphe de la session doit lui aussi être cohérent et minimal.

On dit qu'une session Σ est ***compatible*** avec un scénario S si et seulement s'il existe au moins une solution compatible avec les deux graphes, ce qui revient à la cohérence du graphe résultant de la fusion de G et de Γ. Trois cas se présentent. Si les deux graphes n'ont aucun nœud commun, ils sont ***séparément compatibles***. S'ils ont certains nœuds communs, ils sont ***partiellement compatibles*** et enfin si tous les nœuds du scénario sont présents dans la session, la ***compatibilité est totale***. Pour une session et un scénario simples, il suffit que leur intersection soit cohérente pour avoir la compatibilité. Si la session est simple et le scénario quelconque, il y a compatibilité si $G(S) \cap \Gamma(\Sigma)$ est un graphe cohérent inclus dans au moins un composant cohérent de G(S). La propriété de compatibilité n'implique pas que la séquence des événements dont la session constitue une trace vérifie effectivement le scénario.

On introduit aussi la notion de satisfaction, plus forte que la compatibilité. Il y a une certaine gradualité dans la notion de satisfaction et on donne 4 formes de satisfaction. La ***satisfaction simple*** est la notion la plus forte. La session Σ satisfait simplement le scénario S si et seulement si pour toute solution σ de $\Gamma(\Sigma)$, il existe au moins une solution s de G(S) compatible avec σ. On est certain que la séquence des événements tels qu'ils se sont effectivement déroulés pendant la session et dont on n'a qu'une description incomplète ou imprécise, vérifie les contraintes imposées par le scénario. De même que pour la compatibilité, on peut nuancer la notion de satisfaction simple en ajoutant si elle est ***totale***, ***partielle*** ou ***séparée***.

Il y a ***satisfaction par inclusion*** si et seulement si Σ et S sont compatibles et si la restriction de G(S) à $\mathcal{W}$ est incluse dans celle de Γ(Σ) à $\mathcal{W}$. Dans ces conditions, il se peut, mais rien ne garantit que pour chaque solution σ de Γ(Σ), il existe une solution s de G(S) compatible avec σ.

On a la ***satisfaction par composantes isolées*** en remplaçant dans la définition précédente l'inclusion par une inclusion par composantes isolées. Alors il se peut, mais rien ne le garantit, que pour chaque solution σ d'un sous-graphe isolé de Γ(Σ) il existe une solution s de G(S) compatible avec σ.

Enfin Σ ***satisfait par contraintes*** S si et seulement si Σ et S sont compatibles et toute contrainte de Σ est, soit disjointe du scénario, soit incluse dans la contrainte du scénario correspondante. C'est la forme de satisfaction la plus faible, elle exclut toute propagation de contraintes donc toute déduction d'information.

L'ordre d'importance des différents degrés de satisfaction est donné par :

sat. simple⇒sat. par inclusion⇒sat. par composantes isolées⇒sat. par contraintes

De plus, si Σ est une session simple, la satisfaction simple est équivalente à la satisfaction par inclusion. Si on a en plus $\mathcal{U}_s \subset \mathcal{U}_\sigma$ alors la satisfaction par inclusion donne la satisfaction totale.

L'algorithme de reconnaissance se découpe selon les étapes suivantes :

1. Réception de la totalité des contraintes premières de la session et transformation de celles-ci en graphe.

2. Complétion du graphe par ajout d'arcs étiquetés par la contrainte universelle et suppression des nœuds non contraints.

3. Calcul du graphe minimal de la session.

4. Détermination de la cohérence. Si le graphe ne comporte aucune étiquette vide, on peut poursuivre l'analyse (étape 5), sinon c'est inutile, les informations relatives à la session sont incohérentes.

5. Détermination des restrictions des graphes de la session et du scénario à $\mathcal{W}$.

6. Détermination de la compatibilité. Si card($\mathcal{W}$)≤1, les graphes sont compatibles. Si 1≤card($\mathcal{W}$)<card($\mathcal{U}$s) et si $\Gamma(\Sigma)_{\mathcal{W}} \cap G(S)_{\mathcal{W}}$ est cohérent, alors la session est partiellement compatible avec le scénario. Passer en 7. Si $\mathcal{W}= \mathcal{U}$s et si $\Gamma(\Sigma)_{\mathcal{W}} \cap G(S)_{\mathcal{W}}$ est cohérent, alors la session est totalement compatible. Passer en 7. Sinon la session est incompatible et la reconnaissance est un échec.

7. Détermination de la satisfaction simple. Si card($\mathcal{W}$)≤1, la session satisfait simplement le scénario. Si $\mathcal{W}= \mathcal{U}_s$ et si $\Gamma_{\mathcal{W}} \subset G_{\mathcal{W}}$, la session satisfait totalement le scénario. Sinon étape 8.

8. Détermination de la satisfaction par composantes isolées. Si $\Gamma_{\mathscr{W}} \subset^{pci} G_{\mathscr{W}}$ alors la session satisfait le scénario par composantes isolées. Sinon il n'y a pas satisfaction.

VII.1.3Reconnaissance incrémentale

La reconnaissance ne se fait plus ici à partir de la totalité de la session mais seulement avec une représentation partielle. On considère en fait des sessions Σ_k obtenues après introduction de k contraintes temporelles. La reconnaissance doit être dynamique, il ne s'agit pas seulement de juger de la situation relative d'un scénario et d'une session partielle, mais aussi d'estimer la possibilité de parvenir ultérieurement à une compatibilité ou à une satisfaction totales.

Il existe un prolongement Σ de Σ_k totalement compatible avec S si et seulement si Σ_k est compatible avec S. De même il existe un prolongement qui satisfait totalement S si et seulement si Σ_k est compatible avec S.

Une session Σ_{k+1} est obtenue par ajout d'une contrainte ρ_{k+1} à Σ_k. Le nouveau graphe de session à considérer est $\Gamma(\Sigma_{k+1})=\min(\Gamma(\Sigma_k)\oplus G(\rho_{k+1}))$, il s'obtient par une opération de fusion et une opération de minimalisation. Soit ρ_k une contrainte de la session et r la contrainte du scénario de mêmes extrémités. ρ_k est compatible avec S si et seulement si $E(r)\cap E(\rho_k)\neq\varnothing$ et ρ_k satisfait S si et seulement si $E(\rho_k)\subset E(r)$. Si on part d'une session Σ_k cohérente à laquelle on ajoute une contrainte ρ_{k+1}, alors Σ_{k+1} est cohérente si et seulement si ρ_{k+1} est compatible avec Σ_k. On déduit de ces propriétés un algorithme de reconnaissance général :

Si ρ_{k+1} est incompatible avec Σ_k

Alors session incohérente : arrêt

Sinon $\Gamma(\Sigma_{k+1})=\min(\Gamma(\Sigma_k)\oplus G(\rho_{k+1}))$

Si $\min(\Gamma(\Sigma_{k+1})\oplus G(S))$ est cohérent

Alors Si $\mathscr{W}_{k+1}=\mathscr{U}_s$ la compatibilité est totale

Sinon la compatibilité est partielle

Si $\Gamma(\Sigma_{k+1})_{\mathscr{W}_{k+1}} \subset G(S)_{\mathscr{W}_{k+1}}$

Alors Si $\mathscr{W}_{k+1}=\mathscr{U}_s$ la satisfaction est totale

Sinon il y a satisfaction par inclusion

Sinon Si $\Gamma(\Sigma_{k+1})_{\mathscr{W}_{k+1}} \subset^{pci} G(S)_{\mathscr{W}_{k+1}}$ il y a satisfaction par composantes isolées

Sinon il y a incompatibilité

On peut aussi introduire un graphe aux propriétés intéressantes, le graphe du scénario actualisé S_k issu de S. $G(S_k)$ est obtenu par intersection des graphes de la session partielle et du scénario initial puis par minimalisation. $G(S_k)=\min[G(S)\oplus\Gamma(\Sigma_k)]$. Actualiser le scénario, c'est en donner une version instantanée qui, tenant compte de l'évolution effective du processus incrémental, permet à elle seule de juger de la compatibilité. La compatibilité et la satisfaction d'une session partielle et d'un scénario sont équivalentes respectivement à la compatibilité et à la satisfaction de la session partielle et du scénario actualisé. Si la session est simple, il suffit que le scénario actualisé soit cohérent pour avoir la compatibilité. Soit Σ_k partiellement compatible avec S, alors Σ_{k+1} est compatible avec S si et seulement si ρ_{k+1} est compatible avec le scénario actualisé S_k.

VII.1.4Sessions particulières

A) Sessions datées

Dans certains cas particuliers de sessions, l'algorithme de reconnaissance peut encore se simplifier. Un premier cas est celui où tous les événements observés sont datés. Les contraintes ρ_k sont maintenant de la forme $U_{i_k}-U_0=\lambda_{i_k}$, U_{i_k} variable d'instant de S, U_0 variable référençant l'origine et λ_{i_k} réel. On appelle ce contexte C_1. La satisfaction devient alors synonyme de compatibilité. Soit $r_{0,i(k+1)}$ la relation de $G(S_k)$ d'extrémités U_0 et $U_{i_{(k+1)}}$: Σ_{k+1} satisfait S si et seulement si $\lambda_{i_{(k+1)}} \in E(r_{0,i_{(k+1)}})$. Le cœur de l'algorithme de reconnaissance devient :

Si $\lambda_{i_k} \in E(r_{0,i_k})$

Alors Si k<n-1

Alors satisfaction partielle et $G(S_{k+1})=\min[G(S_k)\oplus G(\rho_{k+1})]$

Sinon satisfaction totale et arrêt

Sinon incompatibilité et arrêt

B) Sessions datées et ordonnées

On peut encore enrichir l'algorithme dans un nouveau contexte C_2 où les contraintes sont ordonnées en plus d'être datées, c'est-à-dire que l'ordre d'introduction incrémentale coïncide avec l'ordre induit par les datations. La compatibilité partielle n'induit plus forcément la possibilité de parvenir à une compatibilité totale, de même pour la satisfaction puisque l'on a toujours l'équivalence entre ces deux notions. Soit Σ_k vérifiant C_2. On appelle extension de Σ_k un prolongement noté Σ_k^E obtenu à partir de Σ_k par adjonction de l'information exprimant que tous les événements instantanés n'ayant pas encore eu d'occurrence ne pourront en avoir une que dans le futur. Les contraintes sont : $(\rho_1,\rho_2,...,\rho_k,\varepsilon_{i1},\varepsilon_{i2},...,\varepsilon_{il})$, avec ε_{ij} telle que $U_j-U_0 \in [\lambda_{i_k},+\infty]$ $\forall$ $U_j \in \mathcal{U}_s-\mathcal{U}_{\Sigma_k}$.

Il existe un prolongement Σ de Σ_k qui satisfait totalement S si et seulement si Σ_k^E est compatible avec S, condition équivalente à $G(S)\oplus\Gamma(\Sigma_k^E)$ cohérent ou S_k^E cohérent. On appelle $H(\Sigma_k)$ le graphe tel que pour tout $U_j\in\mathcal{U}_s\text{-}\mathcal{U}_{\Sigma_k}$, toutes les contraintes élémentaires entre U_j et U_0 soient étiquetées par $[\lambda_{i_k},+\infty]$ et pour tout $U_j\in\mathcal{U}_{\Sigma_k}$, toutes les contraintes entre U_j et U_0 soient universelles. Le graphe de S_{k+1}^E s'obtient par : $G(S_{k+1}^E)=\min[G(S_k^E)\oplus G(\rho_{k+1})\oplus H(\Sigma_{k+1})]$. L'algorithme devient alors :

Si $G(S_{k+1}^E)$ est cohérent

Alors Si $k<n-1$

Alors satisfaction totale toujours possible

Sinon satisfaction totale et arrêt

Sinon satisfaction totale impossible et arrêt

C) Sessions en temps courant

On fait une hypothèse supplémentaire au contexte C_2. Les informations parviennent maintenant au système de reconnaissance en temps courant, leur date d'arrivée correspond à la date de l'événement. Afin d'éviter un processus de reconnaissance indéfini, on peut introduire un délai à partir de l'activation d'un scénario, délai au-delà duquel il est nécessaire de prendre une décision ; la reconnaissance s'arrête de toute façon. On peut recenser en effet 4 types de rejet d'un scénario : par non satisfaction des contraintes (un événement viole les contraintes du scénario), par mauvais ordonnancement (un événement arrive avant un autre), par absence d'information (un événement prédit n'est pas observé), ou par dépassement (délai atteint).

On définit la notion d'agenda. L'agenda A_k est l'ensemble des variables d'instants U telles que $U \in \mathcal{U}_s\text{-}\mathcal{U}_{\Sigma_k}$ et U n'a pas de prédécesseur dans S_k^E qui appartienne à $\mathcal{U}_s\text{-}\mathcal{U}_{\Sigma_k}$. L'agenda regroupe en fait à chaque instant les événements qui pourraient arriver pour satisfaire le scénario S. Si A_k est vide, il n'y a pas de prolongement de Σ_k qui satisfasse le scénario. Dans le cas d'un scénario simple, on considère $M_k=\inf_j[\text{borne-sup}(E(r_{0j}),j/U_j\in A_k]$. Σ_{k+1} admet un prolongement satisfaisant totalement S si et seulement si $U_{i_{(k+1)}}\in A_k$ et $\lambda_{i_{(k+1)}}\in E(r_{0i_{(k+1)}})$ et $\lambda_{i_{(k+1)}}\leq M_k$. M_k représente la date au plus tard du prochain événement. Si le scénario est disjonctif, cette condition est nécessaire mais pas suffisante. On peut aussi décomposer le scénario disjonctif en scenarii simples, avec pour chacun d'eux une valeur limite M_k^j et la valeur limite globale $M_k=\sup_j(M_k^j)$. La condition est alors suffisante mais pas nécessaire.

Le cœur de l'algorithme est donné pour un scénario disjonctif ou simple :

Si $G(S_k^E)$ cohérent

Alors continuer

Sinon non satisfaction et arrêt

$k<n$

$A_k=\emptyset$

$M_k=+\infty$

suc=faux

Pour tout nœud T de $\mathcal{U}_s$-$\mathcal{U}_k$ faire

Si T n'a pas de prédécesseur dans $\mathcal{U}_s$-$\mathcal{U}_k$

Alors ajouter T à A_k

Si $\sup(E(r_{0T}))<M_k$

Alors $M_k=\sup(E(r_{0T}))$

Si T a un successeur

Alors suc=vrai

Si $M_k=+\infty$ et suc=faux

Alors satisfaction totale et arrêt

Sinon

$k=k+1$

attente(M_k) (scrute l'arrivée d'un événement dans le délai M_k et renvoie U_{i_k}

Si non(contrainte reçue)

Alors événement non produit à temps et arrêt

Si $U_{i_k}\in A_k$

Alors Si $U_{i_k}-U_0\in E(r_{0i_k})$

Alors Si $k=n$

Alors satisfaction totale

Sinon satisfaction partielle et continuer

Sinon Cas où :

$U_{i_k}-U_0>\sup(E(r_{0i_k}))$

U_{i_k} survient trop tard

$U_{i_k}-U_0<\inf(E(r_{0i_k}))$

U_{i_k} survient trop tôt

Sinon U_{i_k} survient trop tôt, d'autres événements devant survenir avant et arrêt

$G(S_{k+1}^E)=\min[G(S_k^E)\oplus G(\rho_{k+1})\oplus H_{k+1}]$

VII.2 Gestion des instances de scénarii

Nous avons montré sur un exemple la nécessité d'un mécanisme de duplication pour la reconnaissance. Voici une présentation plus détaillée des conséquences pour la gestion des instances de scenarii. Avant d'intégrer un nouvel élément dans une chronique, il faut impérativement faire un duplicata de l'instance de la chronique. Les différentes instances d'une même chronique sont gérées par un arbre. A chaque fois qu'un événement e est reconnu, on duplique l'instance S à laquelle il s'intègre et on l'ajoute à celle-ci pour créer une nouvelle instance S'=S augmentée de l'événement e qui, dans l'arbre, constitue le nœud fils de S. Lorsque l'on cherche à intégrer un événement dans une chronique, on parcourt son arbre en profondeur d'abord pour trouver toutes les instances en cours qui n'ont pas encore intégré cet élément. Par construction de l'arborescence, si une instance a déjà intégré une occurrence du type de l'événement qui arrive, alors toute sa descendance possède déjà cette occurrence et il est inutile de parcourir cette branche.

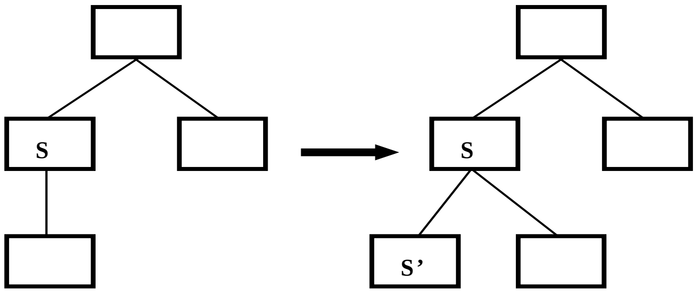


Fig. 134 : duplication d'une instance de chronique

Dans le cas de l'occurrence d'un événement interdit par un modèle de chronique, il n'y a pas de parcours optimal de l'arbre. Il faut examiner chaque instance et lui appliquer l'algorithme de violation d'une assertion par un événement interdit. Si cet algorithme conduit à scinder une instance en deux, on remplace l'instance précédente par deux nouvelles instances sœurs et on leur applique le même traitement.

Lorsqu'une instance meurt parce qu'elle a atteint sa limite de validité, elle est supprimée de l'arbre et ses instances filles sont directement reliées à son instance mère.

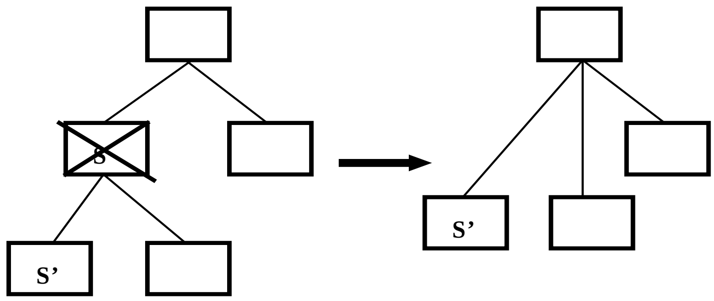


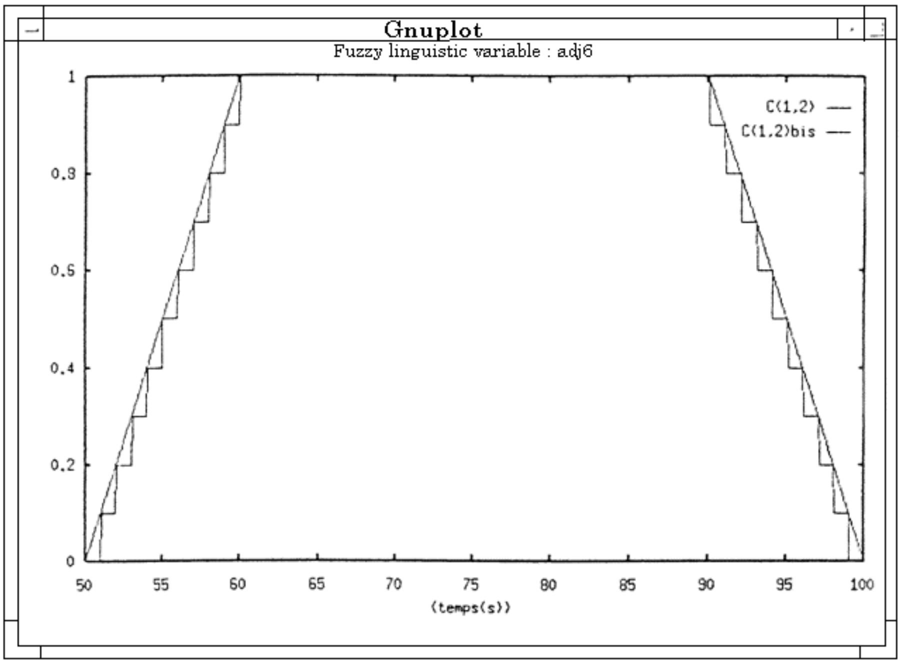
Fig. 135 : mort d'une instance de chronique

On associe à chaque nœud S une date limite qui minore la date de réveil de S et les dates limites de sa descendance. Ceci permet d'optimiser le parcours de l'arbre lors de l'occurrence d'un événement ou d'un tic d'horloge. Pour un tic d'horloge, on parcourt l'arbre en profondeur d'abord en évitant toutes les branches ayant une date de réveil postérieure. On met à jour toutes les instances concernées par cette avance d'horloge et on recalcule les dates limites.

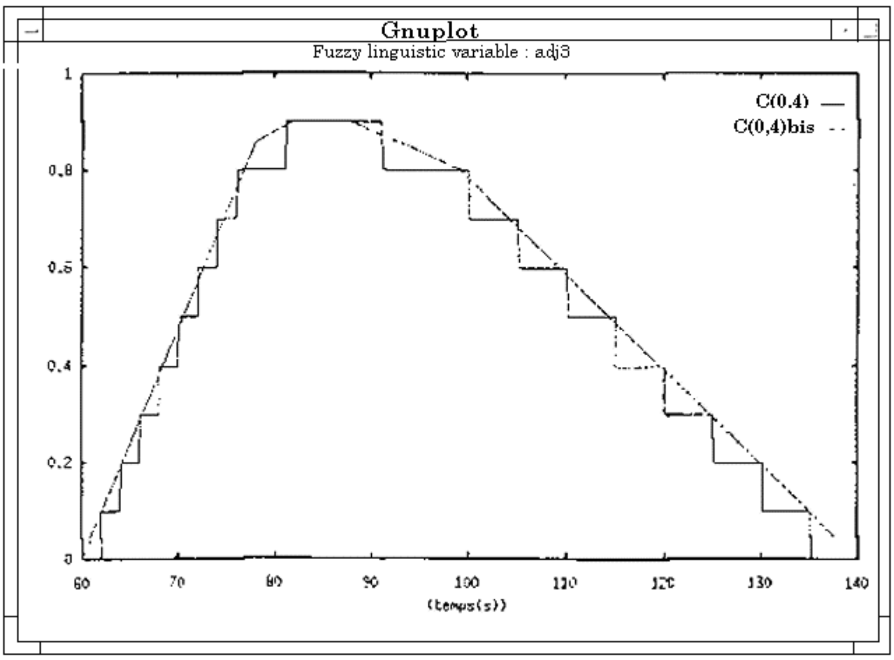
VII.3 Comparaison des différents modèles d'ensembles flous

VII.3.1alpha-coupes / fonctions affines

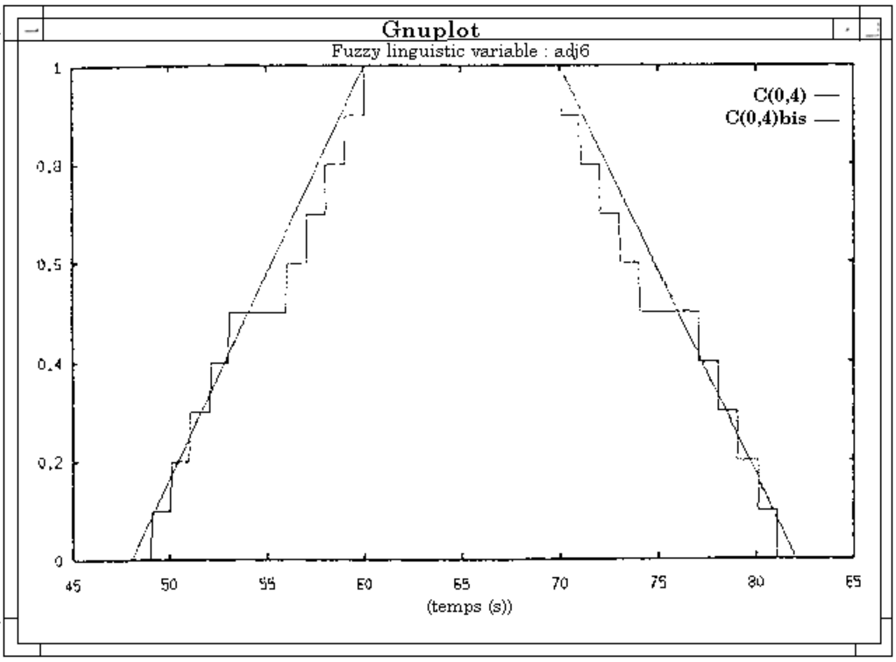
Sur ce résultat, on voit bien que dans certains cas il peut y avoir une équivalence parfaite entre la représentation par α-coupes et par fonctions affines.



Cette équivalence ne se retrouve pas toujours, mais les écarts restent quand-même minimes.

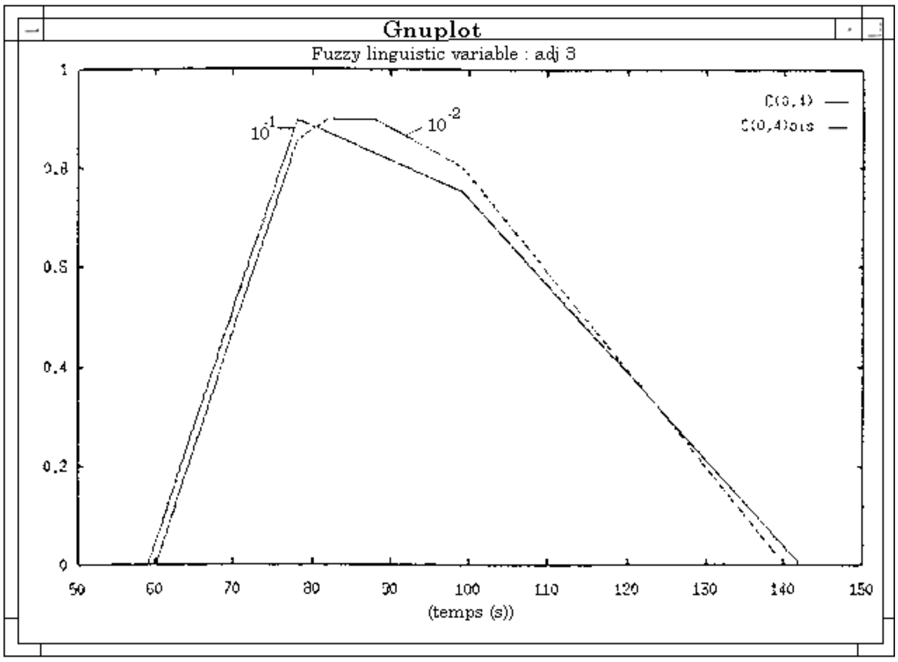


Les écarts deviennent plus importants sur le schéma suivant. On est en effet dans le cas où certaines contraintes initiales sont modélisées de façon grossière. Si on a une contrainte qui s’exprime par C=(0,0,1,1) sous forme trapézoïdale, on devrait la traduire sous forme de α-coupes par C={[-0,9;0,9], [-0,8;0,8], [-0,7;0,7], [-0,6;0,6], [-0,5;0,5], [-0,4;0,4], [-0,3;0,3], [-0,2;0,2], [-0,1;0,1], [0;0]}. Comme on a discrétisé l’univers des temps, C prend en fait pour α-coupes : C={[-1;1], [-1;1], [-1;1], [-1;1], [-1;1], [0;0], [0;0], [0;0], [0;0], [0;0]}. Cet écart initial se répercute de façon imprévisible avec la propagation si on ne modifie pas l’échelle des temps et on peut obtenir ceci :



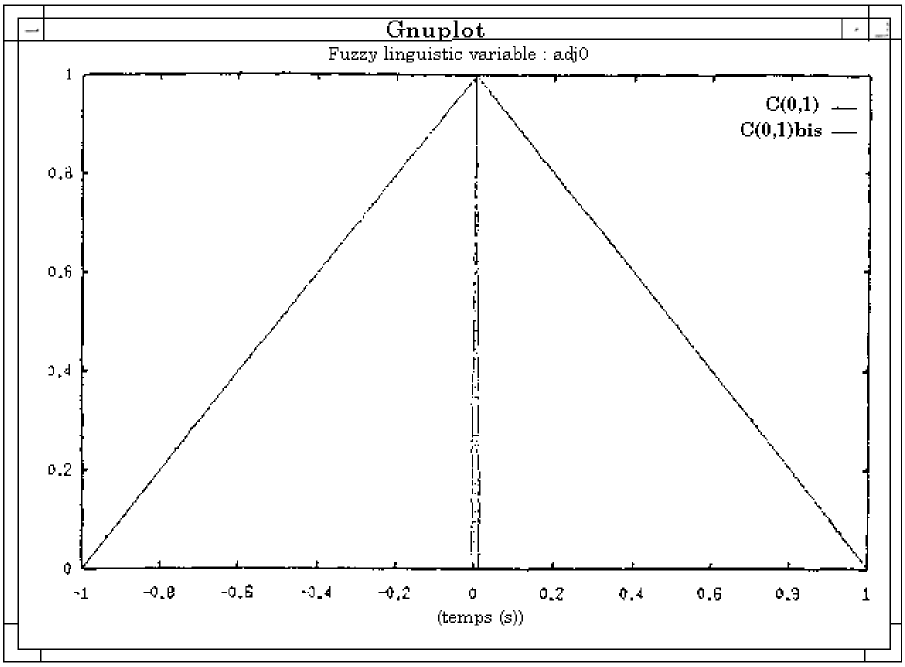
VII.3.2Précision variable

Ici on compare les résultats obtenus sur des contraintes représentées par des fonctions affines en fonctions de la précision des calculs en y. On n'observe aucune différence pour des précisions allant de 10^{-2} à 10^{-5} mais on peut voir sur la figure suivante une légère variation entre une précision de 10^{-1} et de 10^{-2}. Plus la précision est grande, plus on peut avoir de points à calculer.

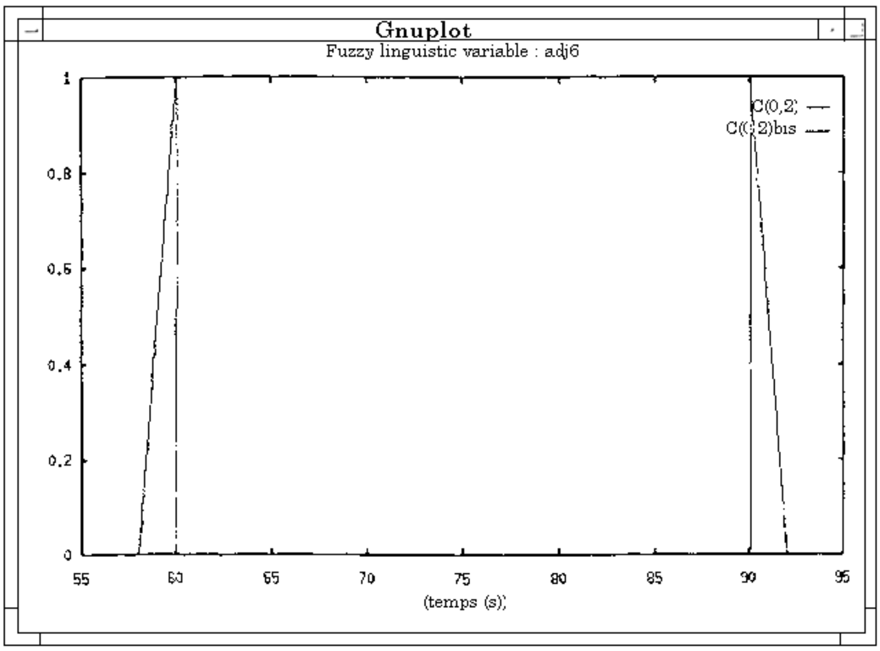


VII.3.3Représentation de contraintes précises

La modélisation par fonctions affines est la moins adaptée pour représenter des contraintes précises du type singleton ou intervalle classique. On ne peut en effet pas représenter de discontinuité. Si on veut représenter un délai exact de 0, on est obligé de l'approcher par le triangle suivant :

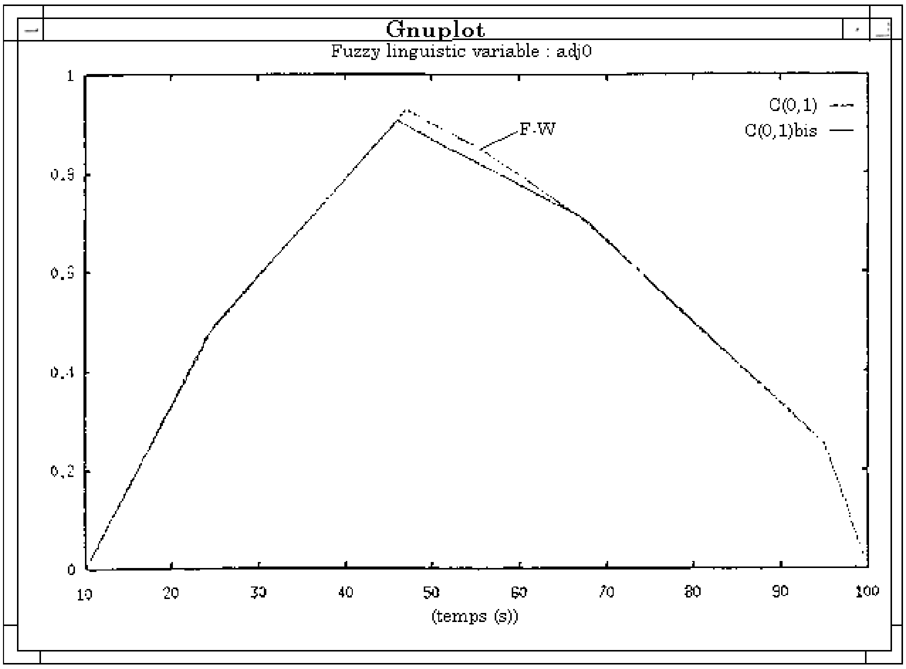


Ceci n'est pas faux puisque pour les valeurs entières -1, 0 et 1 on a bien les degrés d'appartenance souhaités de 0, 1 et 0 respectivement. Mais ceci va introduire une imprécision qui, en se propageant, déforme quelque peu les résultats. Sur cet exemple, le support de la contrainte devrait être [60,90] et on obtient [58,92]. Ce phénomène est négligeable si on considère que les informations sur le terrain ne sont pas données à la seconde près et que, de ce fait, elles sont fournies elles aussi sous la forme d'ensembles flous.



VII.3.4 Comparaison graphes de distances / graphes de contraintes

On a vu que l'application de l'algorithme de Floyd-Warshall sur des distances n'était pas totalement équivalent à celui de Montanari sur des contraintes. Ceci à cause des choix faits pour représenter le graphe. Les différences ne sont quand même pas très importantes. Voici ce que l'on obtient typiquement :



VII.4 Exemple de gabarit et de scénario

Le gabarit qui sert de référence pour les tests est totalement imaginaire et ne correspond à aucune application réelle. On a volontairement construit un gabarit de très grande taille pour que les résultats sur les temps de calcul soient plutôt pessimistes. Chaque nœud de l'arbre correspond à une activité et à deux nœuds dans le graphe de contraintes associé au gabarit : le nœud début de l'activité et le nœud fin de l'activité. Les numéros de ces deux nœuds sont inscrits dans l'arbre et les contraintes sont écrites en fonction de ces numéros. Les contraintes sont décrites sous forme d'intervalles. Ce sont en fait les noyaux des contraintes floues réellement employées.

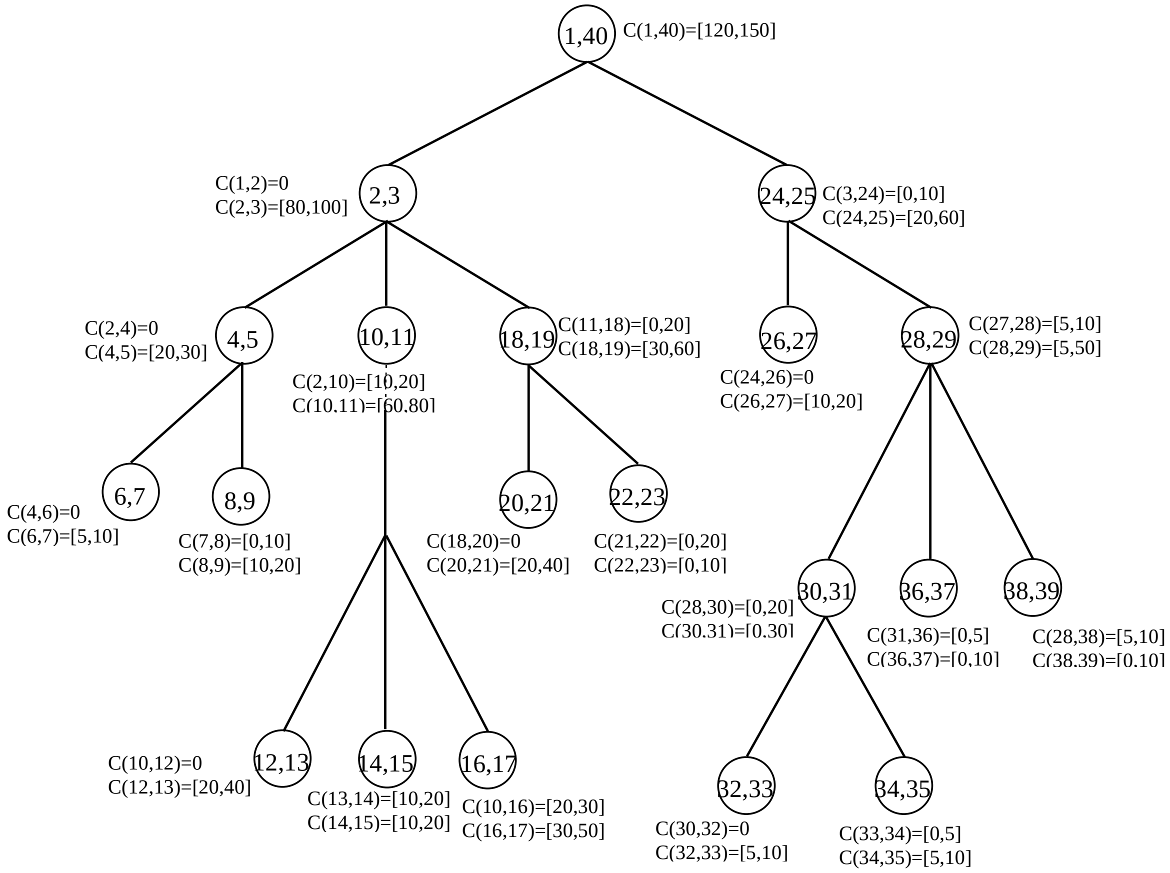


Fig. 136 : exemple de gabarit

On simule sur ce gabarit un scénario en dix étapes. Chaque étape correspond à une observation datée d'un événement et se traduit par deux contraintes, une contrainte sur le début de cet événement et une sur sa fin. Ces contraintes sont décrites ici sous forme de liste de points (abscisse, ordonnée).

étape 1 : C(0,12)={(-∞,1),(600,1),(601,0)} et C(0,13)={(599,0),(600,1),(+∞,1)}

étape 2 : C(0,20)={(-∞,1),(4300,1),(4301,0)} et C(0,21)={(4299,0),(4300,1),(+∞,1)}

étape 3 : C(0,32)={(-∞,1),(6000,1),(6001,0)} et C(0,33)={(5999,0),(6000,1),(+∞,1)}

étape 4 : C(0,34)={(-∞,1),(6500,1),(6501,0)} et C(0,35)={(6499,0),(6500,1),(+∞,1)}

étape 5 : C(0,36)={(-∞,1),(7000,1),(7001,0)} et C(0,37)={(6999,0),(7000,1),(+∞,1)}

étape 6 : C(0,6)={(-∞,1),(-800,1),(-799,0)} et C(0,7)={(-801,0),(-800,1),(+∞,1)}

étape 7 : C(0,8)={(-∞,1),(0,1),(1,0)} et C(0,9)={(-1,0),(0,1),(+∞,1)}

étape 8 : C(0,14)={(-∞,1),(1800,1),(1801,0)} et C(0,15)={(1799,0),(1800,1),(+∞,1)}

étape 9 : C(0,16)={(-∞,1),(600,1),(601,0)} et C(0,17)={(599,0),(600,1),(+∞,1)}

étape 10 : C(0,22)={(-∞,1),(4300,1),(4301,0)} et C(0,23)={(4299,0),(4300,1),(+∞,1)}

VII.5 Comparaison des différents algorithmes

Voici les temps de calcul permettant de comparer les différents algorithmes. Les algorithmes sont appliqués sur le scénario présenté au chapitre précédent. On donne les temps de calcul pour chacune des étapes de 1 à 10, puis la somme de ces temps de calcul (ligne "***Σ***") et enfin le temps de calcul global lorsque les informations des étapes 1 à 10 sont fournies au système en une seule fois (ligne "***1 à 10***").

algo	PC2.0	PC2.1		PC2.2		PC2.3			PC2.4		PC2.5	
p	**/**	**p1**	**p2**	**p1**	**p2**	**/**	**p1**	**p2**	**p1**	**p2**	**p1**	**p2**
1	37	39	35	30	31	105	29	28	26	28	24	25
2	52	49	49	15	16	34	23	21	19	19	18	18
3	30	26	25	23	22	40	35	26	20	21	23	24
4	19	19	17	13	13	14	11	12	13	12	12	12
5	22	18	18	16	16	18	17	17	17	17	17	17
6	19	19	18	11	10	11	11	11	11	11	11	10
7	6	5	6	4	4	5	5	4	4	4	4	4
8	5	5	5	6	6	6	6	6	6	6	6	6
9	11	11	11	10	10	10	10	10	11	11	11	10
10	17	13	13	7	8	6	6	6	7	7	8	7
Σ	218	204	197	135	136	249	153	141	134	136	134	133
1 à 10	46	46	42	38	40	47	33	34	42	46	42	42

VII.6 Distributivité de la composition sur l'intersection.

Le second théorème de Montanari repose sur la distributivité de la composition sur l'intersection. Or cette propriété n'est pas vérifiée sur des contraintes disjonctives floues. Voici un contre-exemple de cette distributivité. On choisit trois ensembles flous A, B et C. Seul A est disjonctif. On va montrer que $A \circ (B \cap C) \neq (A \circ B) \cap (A \circ C)$. On rappelle que les ensembles flous disjonctifs sont modélisés par des fonctions affines par morceaux et que pour ce type d'ensembles les opérations se calculent en appliquant les algorithmes présentés en III.3.2B)c) p.95. De plus, l'opération de composition se ramène à une somme floue (cf. III.1.4A) p.79).

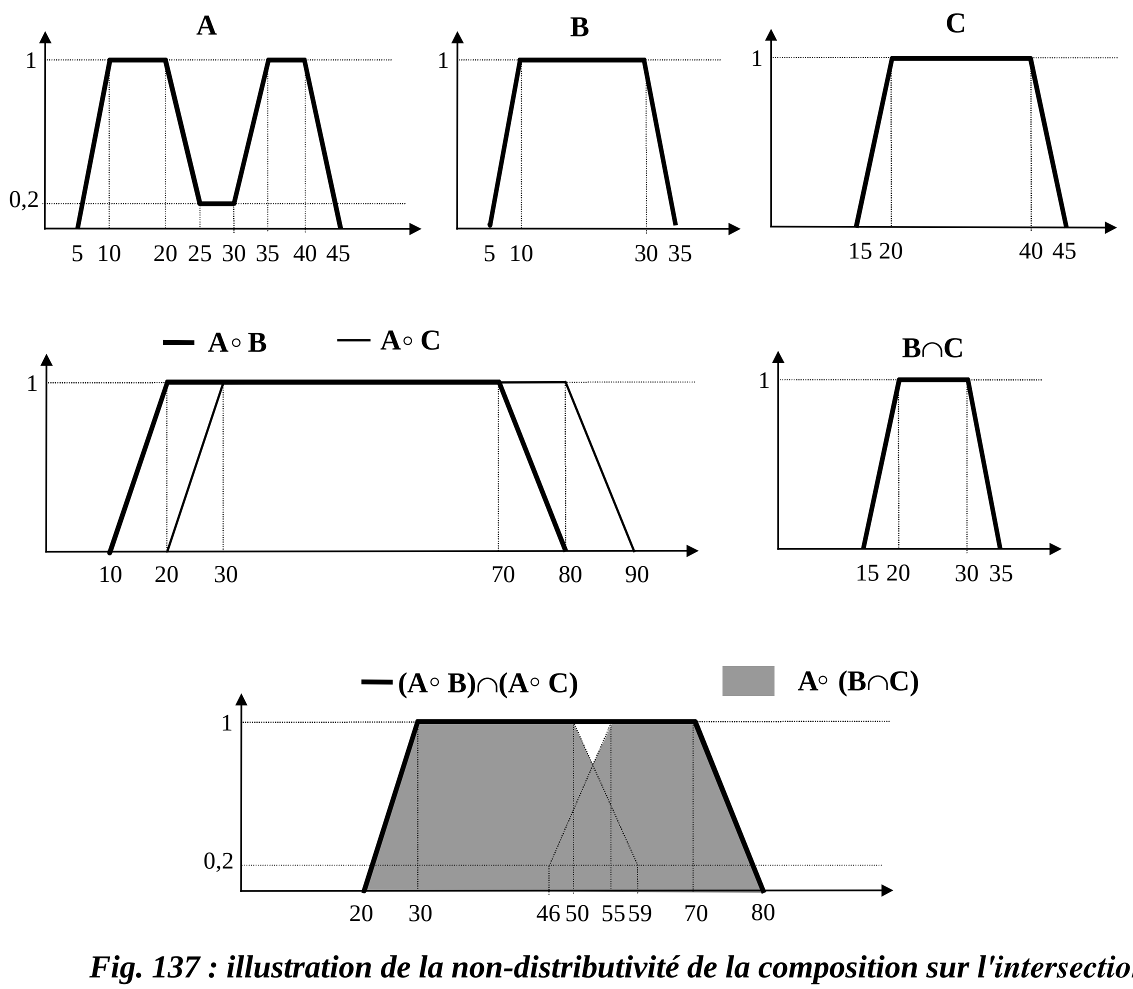


Fig. 137 : illustration de la non-distributivité de la composition sur l'intersection

VII.7 Algorithmes de décomposition des contraintes disjonctives

On considère des ensembles flous modélisés par des listes de points (x_i,y_i) i=1 à n. Un tel ensemble est disjonctif s'il vérifie une des deux conditions suivantes :

Cas 1- $\exists$ i, $1 \le i \le n-2$, / $y_i > y_{i+1}$ et $y_{i+1} < y_{i+2}$

Cas 2- $\exists$ i, $1 \le i \le n-3$, / $y_i > y_{i+1}$ et $y_{i+1} = y_{i+2}$ et $y_{i+2} < y_{i+3}$

Voici un exemple de chacun de ces cas de figure :

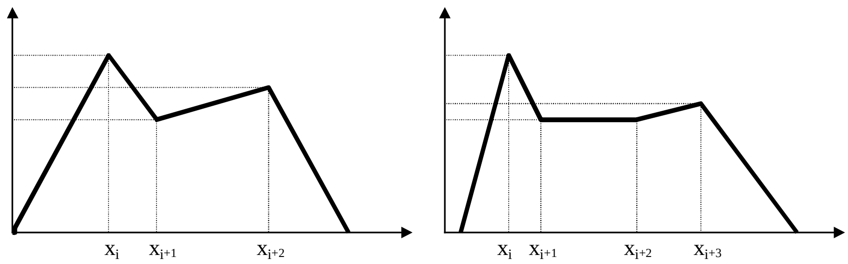


Fig. 138 : exemples de disjonctions

L'algorithme de séparation en contraintes simples distingue ces deux cas. On essaye toujours de minimiser le nombre de points définissant les contraintes simples résultantes.

Cas 1- Si $y_{i+1}=0$, les contraintes sont déjà séparées en deux ensembles (x_j,y_j), j=1 à i+1 et (x_j,y_j), j=i+1 à n.

Sinon on définit les abscisses k_1 et k_2 suivantes :

$$k_1 = \frac{x_{i+2}y_{i+1} - x_{i+1}y_{i+2}}{x_{i+2} - x_{i+1}} \text{ et } k_2 = \frac{x_{i+1}y_i - x_i y_{i+1}}{x_{i+1} - x_i}$$

Les points $(k_1,0)$ et $(k_2,0)$ sont alignés respectivement avec les droites $[(x_{i+1},y_{i+1})(x_{i+2},y_{i+2})]$ et $[(x_i,y_i)(x_{i+1},y_{i+1})]$. Si k_1 et k_2 sont bien à l'intérieur de la contrainte disjonctive, c'est-à-dire $k_1 \ge x_1$ et $k_2 \le x_n$, alors on sépare la contrainte en deux ensembles $\{(x_j,y_j)_{j=1\ldots i},(k_2,0)\}$ et $\{(k_1,0),(x_j,y_j)_{j=i+2\ldots n}\}$. Si $k_2>x_n$, on ne peut pas utiliser le point k_2 et on remplace le premier ensemble par $\{(x_j,y_j)_{j=1\ldots i+1},(x_{i+1}+1,0)\}$, le point $(x_{i+1}+1,0)$ étant obligatoirement à l'intérieur de la contrainte. Ceci oblige à créer une contrainte simple avec i+2 points au lieu de i+1. De même si $k_1<x_1$, on remplace le deuxième ensemble par $\{(x_{i+1}-1,0),(x_j,y_j)_{j=i+1\ldots n}\}$ ce qui fait n-i+1 points au lieu de n-i dans le cas favorable.

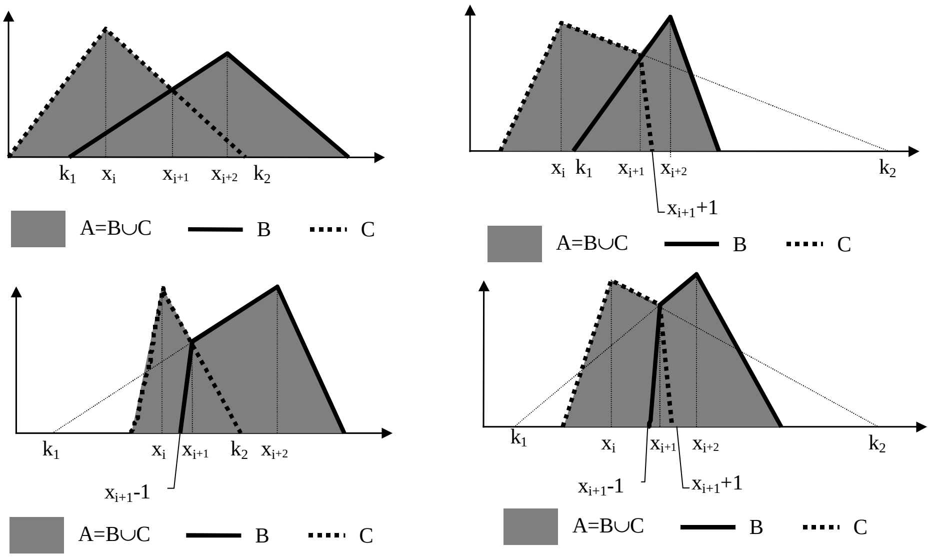


Fig. 139 : séparation pour les disjonctions de cas 1

Cas 2- Si $y_{i+1}=y_{i+2}=0$, les contraintes sont déjà séparées en deux ensembles (x_j,y_j), j=1 à i+1 et (x_j,y_j), j=i+2 à n. Sinon on définit les abscisses k_1 et k_2 suivantes :

$$k_1 = \frac{x_{i+3}y_{i+2} - x_{i+2}y_{i+3}}{x_{i+3} - x_{i+2}} \text{ et } k_2 = \frac{x_{i+2}y_{i+1} - x_{i+1}y_{i+2}}{x_{i+2} - x_{i+1}}$$

Les points $(k_1,0)$ et $(k_2,0)$ sont alignés respectivement avec les droites $[(x_{i+2},y_{i+2})(x_{i+3},y_{i+3})]$ et $[(x_i,y_i)(x_{i+1},y_{i+1})]$. Si k_1 est bien à l'intérieur de la contrainte, c'est-à-dire $k_1 \geq x_1$, on peut utiliser le point $(k_1,0)$ dans les contraintes simples. On peut décomposer la contrainte disjonctive en $\{(x_j,y_j)_{j=1\ldots i+2},(x_{i+2}+1,0)\}$ et $\{(k_1,0), (x_j,y_j)_{j=i+3\ldots n}\}$. Si k_2 est à l'intérieur de la contrainte, on peut aussi décomposer en $\{(x_j,y_j)_{j=1\ldots i},(k_2,0)\}$ et $\{(x_{i+1}-1,0),(x_j,y_j)_{j=i+1\ldots n}\}$. Si ni $(k_1,0)$ ni $(k_2,0)$ ne sont à l'intérieur de la contrainte, on décompose en $\{(x_j,y_j)_{j=1\ldots i+2},(x_{i+2}+1,0)\}$ et $\{(x_{i+1}-1,0),(x_j,y_j)_{j=i+1\ldots n}\}$.

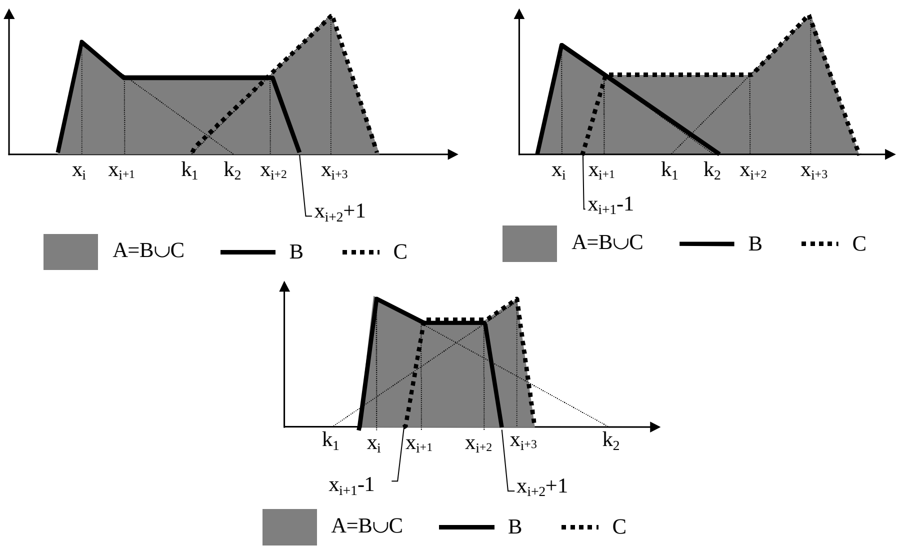


Fig. 140 : séparation pour les disjonctions de cas 2

VII.8 Comportement des opérateurs d'agrégation

Les tableaux suivants illustrent le comportement de certains opérateurs d'agrégation. On a effectué les calculs pour 2 évaluations marginales non pondérées. Les dégradés de gris indiquent où l'on se situe par rapport aux opérateurs à attitude constante de référence. Plus la case est noircie, plus on a une attitude disjonctive.

Signification des couleurs	>max	max	>moy	moy	<moy	min	<min

Opérateurs de sommes symétriques générés par différentes fonctions g

Somme symétrique avec g(x,y)=xy

1,00	0,00	1,00	1,00	1,00	1,00	1,00	1,00
0,80	0,00	0,50	0,73	0,80	0,86	0,94	1,00
0,60	0,00	0,27	0,50	0,60	0,69	0,86	1,00
0,50	0,00	0,20	0,40	0,50	0,60	0,80	1,00
0,40	0,00	0,14	0,31	0,40	0,50	0,73	1,00
0,20	0,00	0,06	0,14	0,20	0,27	0,50	1,00
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Somme symétrique avec g(x,y)=x+y-xy

1,00	0,50	0,56	0,63	0,67	0,71	0,83	1,00
0,80	0,44	0,50	0,56	0,60	0,64	0,73	0,83
0,60	0,38	0,44	0,50	0,53	0,57	0,64	0,71
0,50	0,33	0,40	0,47	0,50	0,53	0,60	0,67
0,40	0,29	0,36	0,43	0,47	0,50	0,56	0,63
0,20	0,17	0,27	0,36	0,40	0,44	0,50	0,56
0,00	0,00	0,17	0,29	0,33	0,38	0,44	0,50
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Somme symétrique avec g(x,y)=min(x,y)

1,00	0,00	1,00	1,00	1,00	1,00	1,00	1,00
0,80	0,00	0,50	0,67	0,71	0,75	0,80	1,00
0,60	0,00	0,33	0,50	0,56	0,60	0,75	1,00
0,50	0,00	0,29	0,44	0,50	0,56	0,71	1,00
0,40	0,00	0,25	0,40	0,44	0,50	0,67	1,00
0,20	0,00	0,20	0,25	0,29	0,33	0,50	1,00
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Somme symétrique avec g(x,y)=max(x,y)

1,00	0,50	0,56	0,63	0,67	0,71	0,83	1,00
0,80	0,44	0,50	0,57	0,62	0,67	0,80	0,83
0,60	0,38	0,43	0,50	0,55	0,60	0,67	0,71
0,50	0,33	0,38	0,45	0,50	0,55	0,62	0,67
0,40	0,29	0,33	0,40	0,45	0,50	0,57	0,63
0,20	0,17	0,20	0,33	0,38	0,43	0,50	0,56
0,00	0,00	0,17	0,29	0,33	0,38	0,44	0,50
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Opérateurs de compensation I^g*U^(1-g) avec I(x,y)=xy et U(x,y)=x+y-xy

g=0,2

1,00	0,00	0,72	0,83	0,87	0,90	0,96	1,00
0,80	0,00	0,60	0,72	0,77	0,81	0,89	0,96
0,60	0,00	0,48	0,60	0,66	0,71	0,81	0,90
0,50	0,00	0,42	0,54	0,60	0,66	0,77	0,87
0,40	0,00	0,36	0,49	0,54	0,60	0,72	0,83
0,20	0,00	0,23	0,36	0,42	0,48	0,60	0,72
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,4

1,00	0,00	0,53	0,69	0,76	0,82	0,91	1,00
0,80	0,00	0,43	0,59	0,65	0,71	0,82	0,91
0,60	0,00	0,34	0,48	0,54	0,60	0,71	0,82
0,50	0,00	0,29	0,42	0,48	0,54	0,65	0,76
0,40	0,00	0,25	0,37	0,42	0,48	0,59	0,69
0,20	0,00	0,15	0,25	0,29	0,34	0,43	0,53
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,5

1,00	0,00	0,45	0,63	0,71	0,77	0,89	1,00
0,80	0,00	0,37	0,53	0,60	0,66	0,78	0,89
0,60	0,00	0,29	0,43	0,49	0,55	0,66	0,77
0,50	0,00	0,24	0,37	0,43	0,49	0,60	0,71
0,40	0,00	0,20	0,32	0,37	0,43	0,53	0,63
0,20	0,00	0,12	0,20	0,24	0,29	0,37	0,45
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,6

1,00	0,00	0,38	0,58	0,66	0,74	0,87	1,00
0,80	0,00	0,31	0,48	0,55	0,62	0,75	0,87
0,60	0,00	0,24	0,38	0,44	0,51	0,62	0,74
0,50	0,00	0,20	0,33	0,39	0,44	0,55	0,66
0,40	0,00	0,17	0,28	0,33	0,38	0,48	0,58
0,20	0,00	0,10	0,17	0,20	0,24	0,31	0,38
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,8

1,00	0,00	0,28	0,48	0,57	0,66	0,84	1,00
0,80	0,00	0,22	0,39	0,47	0,55	0,69	0,84
0,60	0,00	0,17	0,30	0,37	0,43	0,55	0,66
0,50	0,00	0,14	0,26	0,31	0,37	0,47	0,57
0,40	0,00	0,12	0,21	0,26	0,30	0,39	0,48
0,20	0,00	0,06	0,12	0,14	0,17	0,22	0,28
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,9

1,00	0,00	0,23	0,44	0,54	0,63	0,82	1,00
0,80	0,00	0,19	0,35	0,43	0,51	0,67	0,82
0,60	0,00	0,14	0,27	0,33	0,39	0,51	0,63
0,50	0,00	0,12	0,23	0,28	0,33	0,43	0,54
0,40	0,00	0,10	0,18	0,23	0,27	0,35	0,44
0,20	0,00	0,05	0,10	0,12	0,14	0,19	0,23
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Opérateurs FIMICA de Kelman pour différentes valeurs de g

g=0

1,00	1,00	1,00	1,00	1,00	1,00	1,00
0,80	0,80	1,00	1,00	1,00	1,00	1,00
0,60	0,60	0,80	1,00	1,00	1,00	1,00
0,40	0,40	0,60	0,80	1,00	1,00	1,00
0,20	0,20	0,40	0,60	0,80	1,00	1,00
0,00	0,00	0,20	0,40	0,60	0,80	1,00
y/x	**0,00**	**0,20**	**0,40**	**0,60**	**0,80**	**1,00**

g=0,2

1,00	0,80	1,00	1,00	1,00	1,00	1,00
0,80	0,60	0,80	1,00	1,00	1,00	1,00
0,60	0,40	0,60	0,80	1,00	1,00	1,00
0,40	0,20	0,40	0,60	0,80	1,00	1,00
0,20	0,00	0,20	0,40	0,60	0,80	1,00
0,00	0,00	0,00	0,20	0,40	0,60	0,80
y/x	**0,00**	**0,20**	**0,40**	**0,60**	**0,80**	**1,00**

g=0,4

1,00	0,60	0,80	1,00	1,00	1,00	1,00
0,80	0,40	0,60	0,80	1,00	1,00	1,00
0,60	0,20	0,40	0,60	0,80	1,00	1,00
0,40	0,00	0,20	0,40	0,60	0,80	1,00
0,20	0,00	0,00	0,20	0,40	0,60	0,80
0,00	0,00	0,00	0,00	0,20	0,40	0,60
y/x	**0,00**	**0,20**	**0,40**	**0,60**	**0,80**	**1,00**

g=0,6

1,00	0,40	0,60	0,80	1,00	1,00	1,00
0,80	0,20	0,40	0,60	0,80	1,00	1,00
0,60	0,00	0,20	0,40	0,60	0,80	1,00
0,40	0,00	0,00	0,20	0,40	0,60	0,80
0,20	0,00	0,00	0,00	0,20	0,40	0,60
0,00	0,00	0,00	0,00	0,00	0,20	0,40
y/x	**0,00**	**0,20**	**0,40**	**0,60**	**0,80**	**1,00**

g=0,8

1,00	0,20	0,40	0,60	0,80	1,00	1,00
0,80	0,00	0,20	0,40	0,60	0,80	1,00
0,60	0,00	0,00	0,20	0,40	0,60	0,80
0,40	0,00	0,00	0,00	0,20	0,40	0,60
0,20	0,00	0,00	0,00	0,00	0,20	0,40
0,00	0,00	0,00	0,00	0,00	0,00	0,20
y/x	**0,00**	**0,20**	**0,40**	**0,60**	**0,80**	**1,00**

g=1

1,00	0,00	0,20	0,40	0,60	0,80	1,00
0,80	0,00	0,00	0,20	0,40	0,60	0,80
0,60	0,00	0,00	0,00	0,20	0,40	0,60
0,40	0,00	0,00	0,00	0,00	0,20	0,40
0,20	0,00	0,00	0,00	0,00	0,00	0,20
0,00	0,00	0,00	0,00	0,00	0,00	0,00
y/x	**0,00**	**0,20**	**0,40**	**0,60**	**0,80**	**1,00**

Somme g-symétrique g1 pour diffférentes valeurs de g

g=0,2

1,00	0,28	0,33	0,39	0,43	0,48	0,63	0,91
0,80	0,26	0,30	0,35	0,38	0,41	0,50	0,63
0,60	0,23	0,26	0,30	0,33	0,35	0,41	0,48
0,50	0,22	0,25	0,29	0,31	0,33	0,38	0,43
0,40	0,20	0,23	0,27	0,29	0,30	0,35	0,39
0,20	0,17	0,20	0,23	0,25	0,26	0,30	0,33
0,00	0,13	0,17	0,20	0,22	0,23	0,26	0,28
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,4

1,00	0,44	0,49	0,54	0,58	0,62	0,72	0,85
0,80	0,40	0,44	0,50	0,53	0,56	0,63	0,72
0,60	0,36	0,40	0,45	0,47	0,50	0,56	0,62
0,50	0,33	0,38	0,42	0,45	0,47	0,53	0,58
0,40	0,30	0,35	0,40	0,42	0,45	0,50	0,54
0,20	0,24	0,30	0,35	0,38	0,40	0,44	0,49
0,00	0,17	0,24	0,30	0,33	0,36	0,40	0,44
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,6

1,00	0,56	0,60	0,64	0,67	0,70	0,76	0,83
0,80	0,51	0,56	0,60	0,62	0,65	0,70	0,76
0,60	0,46	0,50	0,55	0,58	0,60	0,65	0,70
0,50	0,42	0,48	0,53	0,55	0,58	0,62	0,67
0,40	0,38	0,44	0,50	0,53	0,55	0,60	0,64
0,20	0,28	0,37	0,44	0,48	0,50	0,56	0,60
0,00	0,15	0,28	0,38	0,42	0,46	0,51	0,56
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,8

1,00	0,72	0,74	0,77	0,78	0,80	0,83	0,87
0,80	0,67	0,70	0,74	0,75	0,77	0,80	0,83
0,60	0,61	0,65	0,70	0,71	0,73	0,77	0,80
0,50	0,57	0,62	0,67	0,69	0,71	0,75	0,78
0,40	0,52	0,59	0,65	0,67	0,70	0,74	0,77
0,20	0,37	0,50	0,59	0,62	0,65	0,70	0,74
0,00	0,09	0,37	0,52	0,57	0,61	0,67	0,72
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0.95

1,00	0,91	0,92	0,93	0,93	0,94	0,95	0,95
0,80	0,89	0,90	0,91	0,92	0,92	0,94	0,95
0,60	0,86	0,88	0,90	0,90	0,91	0,92	0,94
0,50	0,83	0,86	0,89	0,90	0,90	0,92	0,93
0,40	0,80	0,84	0,88	0,89	0,90	0,91	0,93
0,20	0,67	0,79	0,84	0,86	0,88	0,90	0,92
0,00	0,02	0,67	0,80	0,83	0,86	0,89	0,91
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Somme g-symétrique g2 pour diffférentes valeurs de g

g=0,2

y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**
1,00	0,25	0,27	0,29	0,30	0,32	0,36	0,40
0,80	0,23	0,25	0,28	0,29	0,31	0,35	0,36
0,60	0,22	0,24	0,27	0,28	0,30	0,31	0,32
0,50	0,21	0,23	0,26	0,28	0,28	0,29	0,30
0,40	0,20	0,22	0,25	0,26	0,27	0,28	0,29
0,20	0,18	0,20	0,22	0,23	0,24	0,25	0,27
0,00	0,15	0,18	0,20	0,21	0,22	0,23	0,25

g=0,4

y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**
1,00	0,44	0,48	0,53	0,56	0,60	0,69	0,80
0,80	0,40	0,44	0,50	0,53	0,57	0,67	0,69
0,60	0,36	0,40	0,46	0,49	0,53	0,57	0,60
0,50	0,33	0,37	0,43	0,47	0,49	0,53	0,56
0,40	0,30	0,34	0,40	0,43	0,46	0,50	0,53
0,20	0,23	0,27	0,34	0,37	0,40	0,44	0,48
0,00	0,13	0,23	0,30	0,33	0,36	0,40	0,44

g=0,6

y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**
1,00	0,55	0,60	0,65	0,69	0,72	0,80	0,90
0,80	0,50	0,55	0,60	0,63	0,67	0,75	0,80
0,60	0,44	0,48	0,53	0,56	0,60	0,67	0,72
0,50	0,40	0,44	0,49	0,53	0,56	0,63	0,69
0,40	0,36	0,40	0,45	0,49	0,53	0,60	0,65
0,20	0,27	0,30	0,40	0,44	0,48	0,55	0,60
0,00	0,15	0,27	0,36	0,40	0,44	0,50	0,55

g=0,8

y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**
1,00	0,67	0,71	0,75	0,78	0,80	0,85	0,91
0,80	0,62	0,66	0,70	0,72	0,75	0,80	0,85
0,60	0,56	0,60	0,64	0,66	0,69	0,75	0,80
0,50	0,53	0,57	0,61	0,63	0,66	0,72	0,78
0,40	0,50	0,53	0,57	0,61	0,64	0,70	0,75
0,20	0,43	0,46	0,53	0,57	0,60	0,66	0,71
0,00	0,34	0,43	0,50	0,53	0,56	0,62	0,67

g=1

y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**
1,00	0,80	0,83	0,87	0,89	0,91	0,95	1,00
0,80	0,75	0,78	0,82	0,84	0,86	0,90	0,95
0,60	0,70	0,73	0,76	0,78	0,80	0,86	0,91
0,50	0,67	0,70	0,73	0,75	0,78	0,84	0,89
0,40	0,64	0,67	0,70	0,73	0,76	0,82	0,87
0,20	0,57	0,60	0,67	0,70	0,73	0,78	0,83
0,00	0,50	0,57	0,64	0,67	0,70	0,75	0,80

Somme g-symétrique g3 pour diffférentes valeurs de g

g=0,2

1,00	0,72	0,87	1,00	1,00	1,00	1,00	1,00
0,80	0,54	0,70	0,85	0,92	0,99	1,00	1,00
0,60	0,37	0,54	0,70	0,77	0,85	0,99	1,00
0,50	0,28	0,45	0,61	0,69	0,77	0,92	1,00
0,40	0,20	0,37	0,53	0,61	0,70	0,85	1,00
0,20	0,03	0,20	0,37	0,45	0,54	0,70	0,87
0,00	0,00	0,03	0,20	0,28	0,37	0,54	0,72
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,4

1,00	0,56	0,71	0,86	0,92	0,98	1,00	1,00
0,80	0,40	0,56	0,70	0,78	0,84	0,97	1,00
0,60	0,24	0,40	0,55	0,63	0,70	0,84	0,98
0,50	0,17	0,32	0,48	0,55	0,63	0,78	0,92
0,40	0,10	0,25	0,40	0,48	0,55	0,70	0,86
0,20	0,00	0,10	0,25	0,32	0,40	0,56	0,71
0,00	0,00	0,00	0,10	0,17	0,24	0,40	0,56
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,6

1,00	0,44	0,60	0,76	0,83	0,90	1,00	1,00
0,80	0,29	0,44	0,60	0,68	0,75	0,90	1,00
0,60	0,14	0,30	0,45	0,52	0,60	0,75	0,90
0,50	0,08	0,23	0,37	0,45	0,52	0,68	0,83
0,40	0,02	0,16	0,30	0,37	0,45	0,60	0,76
0,20	0,00	0,03	0,16	0,23	0,30	0,44	0,60
0,00	0,00	0,00	0,02	0,08	0,14	0,29	0,44
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,8

1,00	0,28	0,46	0,63	0,72	0,80	0,97	1,00
0,80	0,13	0,30	0,46	0,55	0,63	0,80	0,97
0,60	0,00	0,15	0,30	0,39	0,47	0,63	0,80
0,50	0,00	0,08	0,23	0,31	0,39	0,55	0,72
0,40	0,00	0,01	0,15	0,23	0,30	0,46	0,63
0,20	0,00	0,00	0,01	0,08	0,15	0,30	0,46
0,00	0,00	0,00	0,00	0,00	0,00	0,13	0,28
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0.95

1,00	0,09	0,28	0,47	0,57	0,66	0,85	1,00
0,80	0,00	0,10	0,29	0,38	0,48	0,66	0,85
0,60	0,00	0,00	0,10	0,20	0,29	0,48	0,66
0,50	0,00	0,00	0,01	0,10	0,20	0,38	0,57
0,40	0,00	0,00	0,00	0,01	0,10	0,29	0,47
0,20	0,00	0,00	0,00	0,00	0,00	0,10	0,28
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,09
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Somme g-symétrique g4 pour diffférentes valeurs de g

g=0,2

1,00	0,75	0,93	1,00	1,00	1,00	1,00	1,00
0,80	0,57	0,75	0,92	1,00	1,00	1,00	1,00
0,60	0,38	0,56	0,73	0,82	0,90	1,00	1,00
0,50	0,29	0,47	0,64	0,73	0,82	1,00	1,00
0,40	0,20	0,38	0,55	0,64	0,73	0,92	1,00
0,20	0,02	0,20	0,38	0,47	0,56	0,75	0,93
0,00	0,00	0,02	0,20	0,29	0,38	0,57	0,75
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,4

1,00	0,56	0,72	0,87	0,94	1,00	1,00	1,00
0,80	0,40	0,56	0,70	0,77	0,83	0,93	1,00
0,60	0,24	0,40	0,54	0,61	0,67	0,83	1,00
0,50	0,17	0,33	0,47	0,53	0,61	0,77	0,94
0,40	0,10	0,26	0,40	0,47	0,54	0,70	0,87
0,20	0,00	0,13	0,26	0,33	0,40	0,56	0,72
0,00	0,00	0,00	0,10	0,17	0,24	0,40	0,56
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,6

1,00	0,45	0,60	0,75	0,81	0,88	1,00	1,00
0,80	0,30	0,45	0,60	0,67	0,73	0,85	1,00
0,60	0,16	0,32	0,47	0,54	0,60	0,73	0,88
0,50	0,10	0,26	0,41	0,48	0,54	0,67	0,81
0,40	0,04	0,20	0,35	0,41	0,47	0,60	0,75
0,20	0,00	0,10	0,20	0,26	0,32	0,45	0,60
0,00	0,00	0,00	0,04	0,10	0,16	0,30	0,45
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=0,8

1,00	0,33	0,49	0,65	0,72	0,80	0,95	1,00
0,80	0,18	0,34	0,50	0,58	0,65	0,80	0,95
0,60	0,04	0,20	0,36	0,44	0,51	0,65	0,80
0,50	0,00	0,13	0,29	0,37	0,44	0,58	0,72
0,40	0,00	0,07	0,23	0,29	0,36	0,50	0,65
0,20	0,00	0,00	0,07	0,13	0,20	0,34	0,49
0,00	0,00	0,00	0,00	0,00	0,04	0,18	0,33
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

g=1

1,00	0,20	0,37	0,53	0,61	0,69	0,85	1,00
0,80	0,05	0,22	0,38	0,46	0,54	0,70	0,85
0,60	0,00	0,07	0,24	0,32	0,40	0,54	0,69
0,50	0,00	0,00	0,17	0,25	0,32	0,46	0,61
0,40	0,00	0,00	0,10	0,17	0,24	0,38	0,53
0,20	0,00	0,00	0,00	0,00	0,07	0,22	0,37
0,00	0,00	0,00	0,00	0,00	0,00	0,05	0,20
y/x	**0,00**	**0,20**	**0,40**	**0,50**	**0,60**	**0,80**	**1,00**

Acronymes

APU **A**uxiliary **P**ower **U**nit

ATMS **A**ssumption **T**ruth **M**aintenance **S**ystem

C^{3}I **C**ommande, **C**ontrôle, **C**ommunication et **I**ntelligence

CP **C**ollision **P**arallelogram

CSP **C**onstraints **S**atisfaction **P**roblem

FCSP **F**uzzy **C**onstraints **S**atisfaction **P**roblem

FIMICA **F**ixed **I**dentity **M**onotonic **I**dentity **C**ommutative **A**ggregator

GPS **G**lobal **P**ositionning **S**ystem

IA **I**nterval **A**lgebra

IxTeT **I**nde**x**ed **T**im**e** **T**able

KMP **K**nuth, **M**orris et **P**ratt

MATIS **M**odèle d'**A**ide à la décision pour le **T**raitement de l'**I**nformation **S**ymbolique

MICA **M**onotonic **I**dentity **C**ommutative **A**ggregator

OWA **O**rdered **W**eighted **A**veraging operators

PA **P**oint **A**lgebra

PCT **P**arsimonious **C**overing **T**heory

PERT **P**rogram, **E**valuation and **R**eview **T**echnique

RCC **R**egion **C**onnection **C**alculus

SA **S**ingle **A**lgebra

SCAN **SC**ript-**AN**alyst

SIC **S**ystème d'**I**nformation et de **C**ommandement

SIMICA **S**elf **I**dentity **M**onotonic **I**dentity **C**ommutative **A**ggregator

STP **S**ingle **T**emporal **P**roblem

TCSP **T**emporal **C**onstraints **S**atisfaction **P**roblem

TMM **T**ime **M**ap **M**anager

TSA **T**emporal **S**ystem **A**nalyzer

Table des figures

Bibliographie

[Allen 81] J. F. Allen : "An interval-based representation of temporal knowledge", Proceedings of the 7th IJCAI, 1981, pp. 221-226.

[Allen 84] J. F. Allen : "Toward a general theory of action and time", Artificial Intelligence 23, 1984, pp. 123-154.

[Azarewicz 86] J. Azarewicz & al. : "Plan Recognition for Airborne Tactical Decision Making", Proc. of the 5th AAAI'86, Philadelphia, 1986, pp.805-811.

[Asher 95] N. Asher, L. Vieu : "Toward a geometry of common sense : a semantics and a complete axiomatisation of mereotopology", Proceedings of IJCAI'95, pp. 846-852, 1995.

[Barès 94] M. Barès et al. : "Xplans : Case-based Reasoning for Plan Recognition", Applied Artificial Intelligence vol 8 n°4, 1994, pp.617-643.

[Barro 94] S. Barro, R. Marin, J. Mira, A. R. Paton : "A model and a language for the fuzzy representation and handling of time", Fuzzy Sets and Systems n°61, 1994, pp. 153-175.

[Basseville 96] M. Basseville, M-O. Cordier : "Surveillance et diagnostic de systèmes dynamiques : approches complémentaires du traitement de signal et de l'intelligence artificielle", Technical Report 1004, IRISA, Mars 1996.

[Bennett 96] B. Bennett : "Carving up space : steps towards construction of an absolutely complete theory of spatial regions", Logics in Artificial Intelligence, Proceedings of JELIA'96, Lecture Notes in Computer Science n°1126, pp. 337-353, 1996.

[Bestougeff 89] H. Bestougeff, G. Ligozat : "Outils logiques pour le traitement du temps", Masson, 1989.

[Bloch 95] I. Bloch : "Fuzzy geodesic distance in images", IJCAI'95, Lecture Notes in Artificial Intelligence n° 1188, pp. 153-166, 1996.

[Bloch 96.1] I. Bloch : "Information Combination Operators for Data Fusion : a Comparative Review with Classification", IEEE trans. on Systems, Man and Cybernetics vol 26 n°1, Janv. 1996, pp. 52-67.

[Bloch 96.2] I. Bloch : "Fuzzy relative position between objects in images : a morphological approach", International Conference on Image Processing ICIP'96, pp. 987-990, 1996.

[Bloch 97] I. Bloch : "On fuzzy distances and their use in image processing under imprecision", Rapport interne 97D012 de l'Ecole Nationale Supérieure des Télécommunications, 1997.

[Bloch&al 97] I. Bloch, H. Maître, M. Anvari : "Fuzzy adjacency between image objects", International Journal of Uncertainty, Fuzziness and Knowledge-based Systems 5(6), 1997.

[Bloch 98] I. Bloch : "On links between fuzzy morphology and fuzzy distances : euclidean and geodesic cases", International Conference on Information Processing and Management of Uncertainty IPMU'98, Paris, pp. 1144-1151, 1998.

[Bouchon 94] B. Bouchon-Meunier : "La logique floue", Que sais-je ?, Presses Universitaires de France, Avril 1994.

[Bouchon 98] B. Bouchon-Meunier : "Aggregation and fusion of imperfect information", Physica Verlag, 1998.

[Cañamero 94] D. Cañamero : "Modeling Plan Recognition for Decision Support", Future for Knowledge Acquisition, 8th European Knowledge Acquisition Workshop, EKAW'94, pp.158-177.

[Chanas 81] S. Chanas, J. Kamburowski : "The use of fuzzy variables in PERT", Fuzzy Sets and Systems vol 5, 1981, pp. 11-19.

[Chang&al 89] "The design of pictorial databases based upon the theory of symbolic projections", Symposium on the design and implementation of large spatial databases, Lecture Notes in Computer Science n° 409, pp. 303-323, 1989.

[Chauvin 95] S. Chauvin : "Fusion de données par les ensembles flous appliquée à l'imagerie satellitaire : apprentissage et opérateurs", LFA'95, Paris, Décembre 1995, pp. 65-72.

[Chen 93] Z. Chen : "Représentation et gestion de connaissances temporelles et incertaines", Thèse de l'Université Paris XI Orsay, Avril 1993.

[Clementini&al 94.1] E. Clementini, P. Di Felice, P. van Oosterom : "A small set of formal topological relationships suitable for end-user interaction", 3rd International Symposium on Large Spatial Databases SSD'94, Lecture Notes in Computer Science n°692, pp. 277-295, 1994.

[Clementini&al 94.2] E. Clementini, P. Di Felice : "An algebraic model for spatial objects with undetermined boundaries", Proceedings of GISDATA specialist meeting on geographical entities with undetermined boundaries, 1994.

[Clementini&al 95] E. Clementini, P. Di Felice : "A comparison of methods for representing topological relationships", Information sciences n°3, pp. 149-178, 1995.

[Clementini&al 96] E. Clementini, P. Di Felice : "A model for representing topological relationships between complex geometric features in spatial databases", Information sciences 90, pp. 121-136, 1996.

[Clementini&al 97] E. Clementini, P. Di Felice : "A global framework for qualitative shape description", Geoinformatica 1(1), pp. 1-13, 1997.

[Cobb&al 98] M. Cobb, F. E. Petry : "Modeling spatial relationships within a fuzzy framework", Journal of the American Society for Information Science 49(3), pp. 253-266, 1998.

[Cohn&al 93] A. G. Cohn, D. A. Randell, Z. Cui, B. Bennett : "Qualitative spatial reasoning and representation", Qualitative Reasoning and Decision Technologies, CIMNE, Barcelone, 1993.

[Cohn&al 96.1] A. G. Cohn, N. M. Gotts : "The 'Egg-Yolk' representation of regions with indeterminate boundaries", Proceedings of GISDATA Specialist Meeting on Spatial Objects with Undetermined Boundaries, pp171-187, 1996.

[Cohn&al 96.2] A. G. Cohn, N. M. Gotts : "Representing spatial vagueness : a mereological approach", Principle of knowledge representation and reasoning, Proceedings of the 5th International Conference KR'96, pp. 230-241, 1996.

[Cohn&al 97] A. G. Cohn, B. Bennett, J. Gooday, N. M. Gotts : "Qualitative spatial representation and reasoning with the region connection calculus", Geocomputation, pp. 1-44, 1997.

[Collain 95] E. Collain : "Technique des Gabarits", Rapport interne Thomson, RCC/DT/SES/15-233/EC, 1995.

[Cui&al 92] Z. Cui, A. G. Cohn, D. A. Randell : "Qualitative simulation based on a logic of space and time" 6th International Workshop on Qualitative Reasoning about Physical Systems, Herriot-Watt University, Edinburgh, 1992.

[Cui&al 93] Z. Cui, A. G. Cohn, D. A. Randell : "Qualitative and topological relationship", Advances in Spatial Databases, Lecture Notes in Computer Science n°692, pp. 296-315, 1993.

[Davis 95] E. Davis : "A highly expressive language of spatial constraints", Technical Report TR1995-714, Courant Institute, New York, NY, Dec. 1995.

[Dechter 91] R. Dechter, I. Meiri, J. Pearl : "Temporal constraint networks", Artificial Intelligence n°49, 1991, pp. 61-95.

[Dojat 98] M. Dojat, N. Ramaux, D. Fontaine : "Scenario recognition for temporal reasoning in medical domain", Special Issue of the journal of Artificial Intelligence in Medecine, 1998, to appear.

[Dousson 94.1] C. Dousson, M. Ghallab : "Suivi et reconnaissance de chroniques", Revue d'IA, vol 8 n°1, 1994, pp 29-61.

[Dousson 94.2] C. Dousson : "Suivi d'évolutions et reconnaissances de chroniques", Thèse du LAAS,Sept. 1994.

[Dubois 78] D. Dubois, H. Prade : "Algorithmes de plus courts chemins pour traiter des données floues", RAIRO Recherche Opérationnelle vol 12 n°2, Mai 1978, pp. 213-227.

[Dubois 80] D. Dubois, H. Prade : "Fuzzy Sets and Systems. Theory and Applications", Mathematics in Science and Engineering vol 144, Academic Press, 1980, pp.36-66.

[Dubois 86] D. Dubois, H. Prade, C. Testemale : "Fuzzy pattern matching with extended capabilities : proximity notions, importance assessment, random sets", Proc. of NAFIPS 86, New Orleans, 1986.

[Dubois 88.1] D. Dubois, H. Prade, C. Testemale : "Weighted fuzzy pattern matching", Fuzzy sets and systems 26, 1988, pp. 313-331.

[Dubois 88.2] D. Dubois, H. Prade : "Possibility Theory : an approach to computerized processing of uncertainty", Plenum Press, 1988.

[Dubois 89] D. Dubois, H. Prade : "Processing fuzzy temporal knowledge", IEEE transactions on systems, man and cybernetics, vol 19 n°4, Juil./Août 1989, pp. 729-744.

[Dubois 91] D. Dubois, J. Lang, H. Prade : "Timed possibilistic logic", Fundamenta informaticae, special issue on AI, 1991.

[Dubois 93] D. Dubois, H. Prade : "Processing fuzzy temporal knowledge", Readings in Fuzzy Sets for Intelligent Systems, 1993, pp. 112-148.

[Dubois 94] D. Dubois, H. Prade : "La fusion d'informations imprécises", Traitement du signal vol 11 n°6, 1994, pp. 447-458.

[Dubois 95] D. Dubois, H. Fargier, H. Prade : "Possibility theory in constraint satisfaction problem : handling priority, preference and uncertainty", Rapport IRIT/93-30-R, CNRS (URA 1399), Mars 1995.

[Dutta 91.1] S. Dutta : "Topological constraints : a representational framework for approximate spatial and temporal reasoning", 2nd Symposium Advances in Spatial Databases SSD'91, Lecture Notes in Computer Science n° 525, pp. 161-180, 1991.

[Dutta 91.2] S. Dutta : "Approximate spatial reasoning : integrating qualitative and quantitative constraints", International journal of approximate reasoning vol 5, pp. 307-331,1991.

[Egenhofer 91] M. J. Egenhofer : "Reasoning about binary topological relations", Advances in Spatial Databases, 2nd symposium SSD'91, Lecture Notes in Computer Science n°525, pp. 143-160, 1991.

[Egenhofer&al 92] M. J. Egenhofer, K. K. Al-Taha : "Reasoning about gradual changes of topological relationships", Theories and methods of spatio-temporal reasoning in geographical database, International conference GIS, Lecture Notes in Computer Science n°639, pp. 196-219, 1992.

[Eude 97] V. Eude, B. Bouchon-Meunier, E. Collain : "Reconnaissance d'activités à l'aide de graphes temporels flous", LFA'97, Lyon (France), 3-4 Déc. 1997, pp. 91-98.

[Eude 98.1] V. Eude, B. Bouchon-Meunier, E. Collain : "Choix d'opérateurs d'agrégation pour l'évaluation de structures hiérarchisées", IPMU'98, Paris (France), 6-10 Jul. 1998, pp. 778-785.

[Eude 98.2] V. Eude : "Recognition of military activities with fuzzy temporal constraints", INES'98, Vienna (Austria), 17-19 Sept. 1998, pp. 251-256.

[Fall 86] T. C. Fall : "Evidential reasoning with temporal aspects", Proceedings of the 5th AAAI National Conference on AI, Philadelphia, 1986, pp. 891-895.

[Fargier 94] H. Fargier : "Problèmes de satisfaction de contraintes flexibles : application à l'ordonnancement de production", Thèse de l'UPS-IRIT, Juin 1994.

[Fodor 94] J. Fodor, M. Roubens : "Fuzzy preference modelling and multicriteria decision support", Theory and Decision Library, Kluwer Academic Publisher, 1994.

[Fodor 95] J. Fodor, J-L. Marichal, M. Roubens : "Characterization of the Ordered Weighted Averaging Operators", IEEE Trans. on Fuzzy Systems vol 3, 1995, pp. 236-240.

[Fontaine 93] D. Fontaine : "Reconnaissance de scénarios temporels", Rapport interne U.R.A C.N.R.S 817 Heudiasyc, UT Compiègne, Juin 1993.

[Fontaine 98] D. Fontaine, N. Ramaux : "An approach by graphs for the recognition of temporal scenarios", IEEE Transactions on Systems, Man and Cybernetics, Juin 1998.

[Fortemps 95] P. Fortemps : "Traitement optimal des problèmes à contraintes flexibles : ordonnancement et système d'équations linéaires", LFA 95, Paris, Nov. 1995.

[Franck 79] : M. J. Franck : "On the simultaneous associativity of f(x,y) and x+y-f(x,y)", Aequationes Mathematicae, Birkhäuser Verlag, Basel, 19, pp. 194-226, 1979.

[Frank 97] A. U. Frank : "Hierarchical spatial reasoning", submitted to IJGIS, 1997.

[Freksa 92] C. Freksa : "Temporal reasoning based on semi-intervals", Artificial Intelligence 54, pp. 199-227, 1992.

[Gahegan 95] M. Gahegan : "Proximity operators for qualitative spatial reasoning", COSIT'95, Lecture Notes in Computer Science n° 988, pp. 31-44, 1995.

[Galton 96] A. Galton : "Taking dimension seriously in qualitative spatial reasoning", 12th european conference on Artificial Intelligence ECAI'96, 1996.

[Gazdik 83] I. Gazdik : "Fuzzy Network Planning", IEEE Transactions on Reliability 32, 1983, pp. 304-313.

[Ghallab 89.1] M. Ghallab, A. Mounir-Alaoui : "Managing efficiently temporal relations through indexed spanning trees", Proceedings of the 11th IJCAI, 1989, pp. 1297-1303.

[Ghallab 89.2] M. Ghallab : "Représentation et gestion de relations temporelles", Actes du 7ème congrès RFIA, AFCET, Paris, nov. 89, pp. 3-20.

[Ghallab 92] M. Ghallab : "Managing temporal knowledge : a survey on problems and approaches", International Symposium on Artificial Intelligence, Robotics and Automation in Space, Toulouse, Oct. 92.

[Godo 95] L. Godo, L. Vila : "Possibilistic temporal reasoning based on fuzzy temporal constraints", IJCAI'95, 1995.

[Grabish 93] M. Grabish : "On the use of fuzzy integral as a fuzzy connective", Proc. of the 2nd IEEE Conf. On Fuzzy Systems, San Francisco, CA, vol 1 pp. 213-218, 1993.

[Grabish 94] M. Grabisch : "Characterization of fuzzy integrals viewed as aggregation operators", Proc. 3rd IEEE Congr. on Fuzzy Systems, Orlando, USA, 1994.

[Grabisch 95.1] M. Grabisch, H. T. Nguyen and E. A. Walker : "Fundamentals of uncertainty calculi with applications to fuzzy inference", Kluwer Academic Publishers, Dordrecht, 1995.

[Grabish 95.2] M. Grabish : "Fuzzy integral in multicriteria decision making", Fuzzy Sets and Systems, 69, pp. 279-298, 1995.

[Grigni 95] M. Grigni, D. Papadias, C. Papadimitriou : "Topological inference", Proceedings of IJCAI'95, Montreal, pp. 901-906, 1995.

[Guesgen 89] H. W. Guesgen : "Spatial reasoning based on Allen's temporal logic", Technical Report TR-89-049, International Computer Science Institute, Berkeley, 1989.

[Guesgen 94] H. W. Guesgen : "A formal framework for weak constraints satisfaction based on fuzzy sets", Technical Report TR-94-026, International Computer Science Institute, Berkeley, California, June 1998.

[Guesgen 95.1] H. W. Guesgen, A. Philpott : "Heuristics for fuzzy constraint satisfaction", Computer Science Report n° 123, University of Auckland, New Zealand, July 1995.

[Guesgen 95.2] H. W. Guesgen : "Towards hybrid spatial reasoning", Lecture Notes in Artificial Intelligence n° 1188, IJCAI'95, pp. 197-206, 1995.

[Haton 91] Haton & al. : "Le raisonnement en Intelligence Artificielle", interéditions, Paris, 1991.

[Hernandez 93] D. Hernandez : "Maintaining qualitative spatial knowledge", Spatial information theory, a theoretical basis for GIS, European Conference COSIT'93, Lecture Notes in Computer Science n°716, pp. 36-53, 1993.

[Hernandez&al 95] D. Hernandez, E. Clementini,P. Di Felice : "Qualitative distances", Spatial Information Theory, International Conference COSIT'95, Lecture Notes in Computer Science n°988, pp.45-57, 1995.

[Hong&al 95] J. Hong, M. J. Egenhofer, A. U. Frank : "On the robustness of qualitative distance and direction reasoning", Proceedings of Auto-carto 12, pp. 301-310, 1995.

[Hong 93] J. Hong : "Plan Recognition on the Basis of Structural and Temporal Constraints", 2nd European Workshop on Planning EWSP'93, Vadstena, Dec. 1993, pp.131-144.

[Isomoto 97] Y. Isomoto & al : "Data model and fuzzy information retrieval for scenic image database : theoretical extension of a traditional crisp model to fuzzy model", Fuzzy information engineering, 1997, pp. 283-289.

[Jeansoulin 97] R. Jeansoulin : "Geographical constraints revisited through the basic relations of contact, share and order", Proceedings of AAAI'97, Workshop on spatio-temporal reasoning, 1997.

[Jungert 93] E. Jungert : "Symbolic spatial reasoning on object shapes for qualitative matching", Spatial information theory, a theoretical basis for GIS, COSIT'93, Lecture Notes in Computer Science n° 716, pp. 444-462, 1993.

[Kaufmann 91] A. Kaufmann, M. M. Gupta : "Introduction to fuzzy arithmetic : theory and applications", Van Nostrand Reinhold, New York, 1991.

[Kautz 86] H. A. Kautz, J. F. Allen : "Generalized Plan Recognition", Proc. of the AAAI'86, pp.32-37.

[Kautz 91.1] H. A. Kautz, P. B. Ladkin : "Integrating metric and qualitative temporal reasoning", AAAI'91, pp. 241-246, 1991.

[Kautz 91.2] H. A. Kautz : "A Formal Theory of Plan Recognition and its Implementation", Reasoning about plans, 1991, pp.69-126.

[Kelman 95] A. Kelman, R. Yager : "Compatibilité et agrégation, méthode de fusion MOI", LFA 95, Paris, Décembre 1995.

[Kelman 96] A. Kelman : "Modèles flous pour l'agrégation de données et l'aide à la décision", Thèse de l'Université Paris 6, Décembre 1996.

[Lakowski 87] S. J. Lakowski, E. J. Hofmann : "Script-Based Reasoning for Situation Monitoring", Proc. of the AAAI'87, pp.819-823.

[Lapointe&al94] S. Lapointe, R. Proulx : "Fuzzy geometric relations to represent hierarchical spatial information", 10^{th} annual conference on uncertainty in artificial intelligence, UAI'94, Seattle, Washington, 1994.

[Latecki&al 93] L. Latecki, R. Röhrig : "Orientation and qualitative angle for spatial reasoning", Proceedings of the 13^{th} IJCAI, Chambéry, France, pp. 1544-1549, 1993.

[Lévy 95] F. Lévy : "Recognising scenarios : a study", Proc. Workshop on Diagnosis from first Principles DX'95, Oct. 1995.

[Ligozat 97] G. Ligozat : "Generalized intervalls : a guided tour", (submitted), 1997.

[Marin 94] Marin, Barro, Bosch, Mira : "Modeling the representation of time from a fuzzy perspective", Cybernetics and Systems n°25, 1994, pp. 217-231.

[Montanari 74] U. Montanari : "Networks of constraints : fondamental properties and aications to picture processing", Information science vol. 7, pp.95-132, 1974.

[Mc Dermott 82] D. Mc Dermott : "A temporal logic for reasonning about processes and plans", Cognitive Science 6 n°2, Avril-Juin 1982, pp. 101-155.

[Meiri 91] I. Meiri : "Combining qualitative and quantitative constraints in temporal reasoning", Proceedings of the AAAI-1991, Los Angeles, pp. 260-267.

[Mounir-Alaoui 90] A. Mounir-Alaoui : "Raisonnement temporel pour la planification et la reconnaissance de situations", Thèse de l'UPS, Laas-CNRS, Toulouse, 1990.

[Mukerjee&al 90] A. Mukerjee, G. Joe : "A qualitative model for space", Proceedings of AAAI'90, pp. 721-727, 1990.

[Nebel 95] B. Nebel, H-J. Bürckert : "Reasoning about temporal relations : a maximal tractable subclass of Allen's interval algebra", Journal of the ACM vol 42 n°1, pp. 43-66, 1995.

[Nerode 90] A. Nerode : "Some lectures on intuitionistic logic" , Logic and computer science, Lecture Notes in Mathematics n°1429, pp. 12-59, 1990.

[O'Hagan 90] M. O'Hagan : "Using maximum entropy-ordered weighted averaging to construct a fuzzy neuron", Proc. 24th Annual IEEE Asilomar Conferencce on Signals, Systems and Computers, Pacific Grove, Ca, pp. 618-623, 1990.

[Papadias 94] D. Papadias, Y. Theodoris : "Spatial relations, minimum bounding rectangles, and spatial data structures", Technical report KDBSLAB-TR-94-04, Department of computer science and engineering, University of California, 1994.

[Papadias&al 94] D. Papadias, A. U. frank, M. Koubarakis : "Constraint-based reasoning in geographic databases : the case of symbolic arrays", Proceedings of the 2nd ICLP Workshop on Deductive Databases, Italy, 1994.

[PRC-GDR 95] Collectif du projet inter-PRC : "Gestion de l'évolutif et de l'incertain dans une base de connaissances", Actes des 5èmes journées nationales, PRC-GDR IA, Nancy, Fev. 1995, teknea, pp.89-94.

[Rifqi 96] M. Rifqi : "Mesures de comparaison, typicalité et classification d'objets flous : théorie et pratique", Thèse de l'Université Paris 6, Décembre 1996.

[Rosenfeld 79] A. Rosenfeld : "Fuzzy digital topology", Information and Control n° 40, pp. 76-87, 1979.

[Rossazza 90] J-P. Rossazza : "Utilisation de hiérarchies de classes floues pour la représentation de connaissances imprécises et sujettes à exceptions : le système SORCIER", Thèse de l'Université Paul Sabatier, Mai 1990.

[Schwalb 93] E. Schwalb, R. Dechter : "Coping with disjunctions in temporal constraint satisfaction problems", AAAI'93, pp. 127-132, 1993.

[Schwalb 98] E. Schwalb, L. Vila : "Temporal constraints : a survey", Constraints : An International Journal, to appear.

[Silvert 79] W. Silvert : "Symmetric summation : a class of operations on fuzzy sets", IEEE Trans. On Systems, Man and Cybernetics, 9, pp. 659-667, 1979.

[Sinha&al 93] D. Sinha, E. R. Dougherty : "Fuzzification of set inclusion : theory and application", Fuzzy Sets and Systems n° 55, pp. 15-42, 1993.

[Song 91] F. Song, R. Cohen : "Temporal reasoning during plan Recognition", Proc. of the AAAI’91, pp.247-252.

[Steunou 92] S. Steunou : "Un apport méthodologique et formel pour le développement de systèmes à base de connaissance temps réel", Thèse de l’Université Paris VI, Avril 92.

[Topaloglou 94.1] T. Topaloglou : "First-Order theories of approximate space", AAAI'94 Workhop on spatial and temporal reasoning, Seattle, Washington, pp. 47-53, 1994.

[Topaloglou 94.2] T. Topaloglou : "Qualitative reasoning about imprecise spatial information", ACM Workshop on advances in geographic information systems GIS'94, Dec. 1994.

[Topaloglou 96] T. Topaloglou : "Efficient algorithms for qualitative reasoning about imprecise space", Proc. Of the 11th canadian artificial intelligence conference AI'96, Toronto, May 1996.

[Topaloglou 97] T. Topaloglou : "On the representation of spatial information in knowledge bases", Thèse de l'université de Toronto, 1997.

[Tversky 77] A. Tversky : "Features of similarity", Psychological review, 84, pp. 327-352.

[Van Beek 92] P. Van Beek : "Reasoning about qualitative temporal information", Artificial Intelligence n°58, 1992, pp. 297-326.

[Wainer 98] J. Wainer, S. Sandri : "A fuzzy temporal / categorical extension to the Parsimonious Covering Theory", International Conference on Information Processing and Management of Uncertainty IPMU'98, Paris, 6-10 Jul. , pp. 1130-1137, 1998.

[Yager 88] R. Yager : "On Ordered Weighted Averaging Operators in Multi-criteria Decion Making", IEEE Transactions on Systems, Man and Cybernetics 18, pp. 183-190, 1988.

[Yager 91] R. Yager : "Connectives and quantifiers in fuzzy sets", Fuzzy Sets and Systems 40, 1, pp. 39-75, 1991.

[Yager 94] R. R. Yager : "Aggregation operators and fuzzy systems modeling", Fuzzy Sets and Systems 67, pp. 129- 146, 1994.

[Yager 95] R. Yager : "Quantifier guided aggregation using OWA operators", Technical Report #MII-1502 Machine Intelligence Institute, Iona College, New Rochelle, NY, 1995.

[Yager 96] R. Yager, A. Rybalov : "Uninorm agregation operators", Fuzzy Sets and Systems 80, pp. 111-120, 1996.

[Yamada 93] A. Yamada : "Studies on spatial description understanding based on geometric constraints satisfaction", Thèse de l'université de Kyoto, Jan. 1993.

[Zimmermann&al 93] K. Zimmermann, C. Freksa : "Qualitative spatial reasoning using orientation, distance, and path knowledge", Proceeding of the 13th IJCAI, Workshop on spatial and temporal reasoning, Chambéry, France, 1993.

[Zimmermann 93] K. Zimmermann : "Enhancing qualitative spatial reasoning-combining orientation and distance", COSIT'93, Lecture Notes in Computer Science n° 716, pp. 69-76, 1993.

[Zimmermann 95] K. Zimmermann : "Measuring without measures, the Δ-calculus", COSIT'95, Lecture Notes in Computer Science n° 988, pp. 59-67, 1995.